Index

A

Titre	Page	Titre	Page
ACINQ	1	ANYPREVOUT (APO)	15
ADAPTOR SIGNATURE	2	AOPP	16
ADDR	3	API	17
ADDR.DAT	4	ARBRE DE MERKLE	18
ADDRV2	5	ARK	19
ADRESSE DE RÉCEPTION	6	ASIC	20
AJUSTEMENT DE DIFFICULTÉ	7	ASMAP	21
ALGORITHME	8	ASSUME UTXO	22
ANALYSE DE CHAINE	9	ASSUME VALID	23
ANCESTOR MINING	10	ATH (ALL-TIME HIGH)	24
ANCHOR OUTPUTS	11	ATLC	25
ANCHORS.DAT	12	ATOMIC SWAP	26
ANCRAGE BILATÉRAL	13	ATTAQUE DES 51%	27
ANONSETS (ANONYMITY SETS)	14		

B

Titre	Page	Titre	Page
BANLIST.DAT	1	BIP137	61
BANLIST.JSON	2	BIP141	62
BARE-MULTISIG	3	BIP143	63
BASE (ARITHMÉTIQUE)	4	BIP144	64
BASE58CHECK	5	BIP145	65
BATCHED SPENDING	6	BIP147	66
BDK (BITCOIN DEV KIT)	7	BIP148	67
BECH32 ET BECH32M	8	BIP149	68
BERKELEYDB	9	BIP150	69
BIG-ENDIAN	10	BIP151	70
BIP (BITCOIN IMPROVEMENT PROPOSAL)	11	BIP152	71
BIP1	12	BIP155	72
BIP2	13	BIP156	73
BIP8	14	BIP173	74
BIP9	15	BIP322	75
BIP10	16	BIP324	76
BIP11	17	BIP326	77
BIP12	18	BIT	78
BIP13	19	BITCOIN (B MAJUSCULE)	79
BIP14	20	BITCOIN (B MINUSCULE)	80
BIP16	21	BITCOIN CASH (BCH)	81
BIP17	22	BITCOIN-CLI	82
BIP21	23	BITCOIN.CONF	83
BIP22	24	BITCOIN CORE	84

Titre	Page	Titre	Page
BIP23	25	BITCOIND	85
BIP30	26	BITCOIND.PID	86
BIP31	27	BITCOIN FOG	87
BIP32	28	BITCOIN GOLD (BTG)	88
BIP34	29	BITCOIN KNOTS	89
BIP35	30	BITCOIN INQUISITION	90
BIP37	31	BITCOIN JESUS	91
BIP38	32	BITCOIN POOLED MINING (BPM)	92
BIP39	33	BITCOIN QT	93
BIP42	34	BITCOIN SATOSHI VISION (BSV)	94
BIP43	35	BITCOINTALK	95
BIP44	36	BIT GOLD	96
BIP47	37	BITVM	97
BIP49	38	BLK????.DAT	98
BIP50	39	BLKINDEX.DAT	99
BIP61	40	BLKTREE/	100
BIP65	41	BLOC	101
BIP66	42	BLOC CANDIDAT	102
BIP68	43	BLOCKCHAIN	103
BIP70	44	BLOCKS INDEX	104
BIP71	45	BLOCKS/BLK?????.DAT	105
BIP72	46	BLOCKS/INDEX/	106
BIP75	47	BLOCKS/REV?????.DAT	107
BIP78	48	BLOCKSIGNERS	108
BIP84	49	BLOCKSTREAM	109
BIP85	50	BLOCK TEMPLATE	110
BIP86	51	BLOCK WITHHOLDING	111
BIP90	52	BLOOM FILTER	112
BIP91	53	B-MONEY	113
BIP111	54	BOLT	114
BIP112	55	BOUTISME	115
BIP113	56	BRANCH-AND-BOUND	116
BIP118	57	BRANCHE	117
BIP119	58	BRC-20	118
BIP123	59	BTC	119
BIP125	60	BTCPAY SERVER	120

C

Titre	Page	Titre	Page
C	1	COINS/	29
C# (C SHARP)	2	COINSHUFFLE	30
C++	3	COLD WALLET	31
CAHOOTS	4	COMMERÇANT	32
CANAL DE PAIMENT	5	COMMIT	33
CAPACITÉ DE CANAL LIGHTNING	6	COMPACT BLOCK RELAY	34
CASHU	7	COMPATIBILITÉ RÉTROSPECTIVE	35
CET	8	CONCATÉNATION	36

Titre	Page	Titre	Page
CHANNEL FACTORIES	9	CONDENSAT (HASH)	37
CHAINSTATE/	10	CONFIRMATION	38
CHARGE UTILE (PAYLOAD)	11	CONSENSUS	39
CHAUMIAN COINJOIN	12	CONSOLIDATION	40
CHIFFRER (CHIFFREMENT)	13	CONTRAT INTELLIGENT	41
CIBLE DE DIFFICULTÉ	14	COOKIE (.COOKIE)	42
CIOH	15	CORE-LIGHTNING (CLN)	43
CLÉ ÉTENDUE	16	COURBE ELLIPTIQUE	44
CLÉ PRIVÉE	17	COVENANT	45
CLÉ PUBLIQUE	18	CPFP (CHILD PAY FOR PARENT)	46
CLI	19	CPPSRB	47
C-LIGHTNING (CLN)	20	CPU (CENTRAL PROCESSING UNIT)	48
CLONE	21	CRYPTANALYSE	49
CODE DE CHAÎNE	22	CRYPTER	50
CODE DE PAIMENT RÉUTILISABLE	23	CRYPTO-ACTIF	51
COINBASE (TRANSACTION)	24	CRYPTOGRAPHIE	52
COIN CONTROL	25	CRYPTOLOGIE	53
COINJOIN	26	CRYPTOMONNAIE	54
COINJUMBLE	27	CYPHERPUNK	55
COINMUX	28		

D

Titre	Page	Titre	Page
DANDELION	1	DLP (DISCREET LOG PROBLEME)	12
DARKWALLET	2	DNS SEEDS	13
DATABASE/	3	DOS (DENIAL OF SERVICE)	14
DB.LOG	4	DOUBLE DÉPENSE (ATTAQUE)	15
DDOS	5	DRIVECHAIN	16
DEBUG.LOG	6	DUMMY ELEMENT	17
DÉPÔT	7	DUST	18
DGM	8	DUSTING ATTACK	19
DIFFIE-HELLMAN	9	DUST LIMIT	20
DISTRIBUÉ	10	DUSTRELAYFEE	21
DLC (DISCREET LOG CONTRACT)	11		

E

Titre	Page	Titre	Page
ECASH (DAVID CHAUM)	1	EMBRANCHEMENT NATUREL	10
ECASH (XEC)	2	ENDIANNESS	11
ECDH	3	ENTÊTE DE BLOC	12
ECDSA (ELLIPTIC CURVE DIGITAL SIGNATURE ALGORITHM)	4	ENTRÉE (INPUT)	13
ECLAIR	5	ENTROPIE	14

Titre	Page	Titre	Page
ECLIPSE (ATTAQUE)	6	EREBUS (ATTAQUE)	15
ÉCOLE AUTRICHIENNE	7	ESMPPS (EQUALIZED SHARED MAXIMUM PAY PER SHARE)	16
ELECTRUM LIGHTNING	8	ÉTIQUETAGE	17
ELTOO	9	EXPLORATEUR DE BLOC	18

F

Titre	Page	Titre	Page
FARADAY	1	FONCTIONNAIRE	10
FEDIMINT	2	FORCE BRUTE (ATTAQUE)	11
FEE SNIPING	3	FORCED ADDRESS REUSE	12
FEE_ESTIMATES.DAT	4	FORK	13
FERME DE MINAGE	5	FORK (GIT)	14
FIAT	6	FORTH	15
FIBRE (FAST INTERNET BITCOIN RELAY ENGINE)	7	FPPS (FULL PAY PER SHARE)	16
FLAG DAY	8	FRAIS DE TRANSACTION	17
FONCTION DE HACHAGE CRYPTOGRAPHIQUE	9		

G

Titre	Page	Titre	Page
GAP LIMIT	1	GOLDFINGER (ATTAQUE)	6
GENÈSE (BLOC)	2	GOSSIP	7
GETWORK	3	GRAINE (SEED)	8
GIT	4	GUI	9
GO (GOLANG)	5	GUISETTINGS.INI.BAK	10

H

Titre	Page	Titre	Page
HAL FINNEY	1	HASHCASH	6
HALVING	2	HASHRATE	7
HARD FORK	3	HMAC-SHA512	8
HARDWARE WALLET	4	HORODATAGE (TIMESTAMP)	9
HARDWARE WALLET INTERFACE (HWI)	5		

I

Titre	Page	Titre	Page
INBOUND CAPACITY	1	INPUT	5
INDEXES/TXINDEX/	2	IP_ASN.MAP	6
INITIAL BLOCK DOWNLOAD (IBD)	3	ISSUE	7
INDEX (NUMÉRO DE CLÉ)	4		

J

Titre	Page	Titre	Page
JAVA	1	JBOK (PORTEFEUILLE)	3
JAVASCRIPT (NODE.JS)	2		

K

Titre	Page	Titre	Page
KNAPSACK SOLVER	1	KYC (KNOW YOUR CUSTOMER)	2

L

Titre	Page	Titre	Page
LABEL	1	LIGHTNING NETWORK DAEMON (LND)	7
LCB/FT	2	LIQUID NETWORK	8
LDK (LIGHTNING DEV KIT)	3	LITTLE-ENDIAN	9
LEVELDB	4	LOCK (.LOCK)	10
LIGHTNING LABS	5	LOGARITHME DISCRET (PROBLÈME)	11
LIGHTNING NETWORK	6	LOOP	12

M

Titre	Page	Titre	Page
MAGICAL BITCOIN	1	MERKLE BLOCK	14
MAGIC NETWORK	2	MÉTHODE D'ACTIVATION	15
MAJORITÉ ÉCONOMIQUE	3	MÉTHODE GÉOMÉTRIQUE	16
MALLÉABILITÉ (TRANSACTION)	4	MINAGE	17
MAN-IN-THE-MIDDLE (MITM)	5	MINAGE ÉGOÏSTE	18
MAPPER (TO MAP)	6	MINAGE FUSIONNÉ	19
MASF (MINER-ACTIVATED SOFT FORK)	7	MINAGE FUSIONNÉ AVEUGLE	20
MAST (MERKELISED ALTERNATIVE SCRIPT TREE)	8	MINEUR	21
MASTER FINGERPRINT	9	MINISCRIPT	22
MAX_BLOC_SIZE	10	MINITAPSCRIPT	23

Titre	Page	Titre	Page
MEMPOOL	11	MODÈLE DE SCRIPT	24
MEMPOOL.DAT	12	MTP (MEDIAN TIME PAST)	25
MERGE	13		

N

Titre	Page	Titre	Page
NESTED SEGWIT	1	NŒUD SPV (OU NŒUD LÉGER)	8
NETWORK-ADJUSTED TIME (NAT)	2	NONCE	9
NEW YORK AGREEMENT (NYA)	3	NSEQUENCE	10
NLOCKTIME	4	NULL DATA	11
NŒUD	5	NULLDUMMY	12
NŒUD COMPLET	6	NVERSION	13
NŒUD ÉLAGUÉ	7		

O

Titre	Page	Titre	Page
OBOE (OFF-BY-ONE ERROR)	1	OP_GREATERTHANOREQUAL (0XA2)	46
OBSOLÈTE (BLOC)	2	OP_HASH160 (0XA9)	47
OCTET (BYTE)	3	OP_HASH256 (0XAA)	48
OFFCHAIN	4	OP_IF (0X63)	49
ONCHAIN	5	OP_IFDUP (0x73)	50
ONION_PRIVATE_KEY	6	OP_LESSTHAN (0X9F)	51
ONION_V3_PRIVATE_KEY	7	OP_LESSTHANOREQUAL (0XA1)	52
OP_0 (0X00)	8	OP_MAX (0XA4)	53
OP_0NOTEQUAL (0X92)	9	OP_MIN (0XA3)	54
OP_1 (0X51)	10	OP_NEGATE (0X8F)	55
OP_1ADD (0X8B)	11	OP_NIP (0x77)	56
OP_1NEGATE (0X4F)	12	OP_NOP (0X61)	57
OP_1SUB (0X8C)	13	OP_NOT (0X91)	58
OP_2 à OP_16 (0X52 à 0X60)	14	OP_NOTIF (0X64)	59
OP_2DROP (0XD6)	15	OP_NUMEQUAL (0X9C)	60
OP_2DUP (0X6E)	16	OP_NUMEQUALVERIFY (0X9D)	61
OP_2OVER (0X70)	17	OP_NUMNOTEQUAL (0X9E)	62
OP_2ROT (0X71)	18	OP_OVER (0X78)	63
OP_2SWAP (0X72)	19	OP_PICK (0X79)	64
OP_3DUP (0X6F)	20	OP_PUSHDATA1 (0x4C)	65
OP_ABS (0X90)	21	OP_PUSHDATA2 (0x4D)	66
OP_ADD (0X93)	22	OP_PUSHDATA4 (0x4E)	67
OP_BOOLAND (0X9A)	23	OP_RETURN (0X6A)	68
OP_BOOLOR (0X9B)	24	OP_RIPEMD160 (0XA6)	69
OP_CAT (0X7E)	25	OP_ROLL (0X7A)	70
OP_CHECKHASHVERIFY (CHV)	26	OP_ROT (0X7B)	71
OP_CHECKLOCKTIMEVERIFY (0XB1)	27	OP_SHA1 (0XA7)	72
OP_CHECKMULTISIG (0XAE)	28	OP_SHA256 (0XA8)	73

Titre	Page	Titre	Page
OP_CHECKMULTISIGVERIFY (0XAF)	29	OP_SIZE (0X82)	74
OP_CHECKSEQUENCEVERIFY (0XB2)	30	OP_SUB (0X94)	75
OP_CHECKSIG (0XAC)	31	OP_SUCCESS	76
OP_CHECKSIGADD (0XBA)	32	OP_SWAP (0X7C)	77
OP_CHECKSIGVERIFY (0XAD)	33	OP_TOALTSTACK (0X6B)	78
OP_CODESEPARATOR (0XAB)	34	OP_TRUE (0X51)	79
OP_DEPTH (0X74)	35	OP_TUCK (0X7D)	80
OP_DROP (0X75)	36	OP_VER (0X62)	81
OP_DUP (0X76)	37	OP_VERIFY (0X69)	82
OP_ELSE (0X67)	38	OP_WITHIN (0XA5)	83
OP_ENDIF (0X68)	39	OPCODES	84
OP_EQUAL (0X87)	40	ORACLE	85
OP_EQUALVERIFY (0X88)	41	ORPHELIN (BLOC)	86
OP_EVAL	42	OU EXCLUSIF	87
OP_FALSE (0X00)	43	OUTBOUND CAPACITY	88
OP_FROMALTSTACK (0X6C)	44	OUTPUT	89
OP_GREATERTHAN (0XA0)	45	OUTPUT LINKING	90

P

Titre	Page	Titre	Page
P2PK	1	PÉRIODE DE MATURITÉ	22
P2PKH	2	PETIT-BOUTISTE	23
P2P TRANSPORT V2	3	PHRASE DE RÉCUPÉRATION (MNÉMONIQUE)	24
P2MS	4	POOL	25
P2SH	5	POOL DE MINAGE	26
P2SH-P2WPKH	6	POOL HOPPING	27
P2SH-P2WSH	7	PORTE DÉROBÉE (BACKDOOR)	28
P2TR	8	PORTEFEUILLE	29
P2WPKH	9	PORTEFEUILLE CHAUD (LOGICIEL)	30
P2WSH	10	PORTEFEUILLE FROID	31
PAIR-À-PAIR (P2P)	11	POT (PAY ON TARGET)	32
PAIR ENTRANT	12	PPLNS (PAY PER LAST N SHARES)	33
PAIR SORTANT	13	PPLNSG (PAY PER LAST N SHARES GROUPED)	34
PASSPHRASE (BIP39)	14	PPS (PAY PER SHARE)	35
PATOSHI	15	PRÉFIXES BINAIRES	36
PAYJOIN	16	PREUVE DE TRAVAIL	37
PAYNYM	17	PROOF-OF-WORK	38
PBKDF2	18	PROP (PROPORTIONAL)	39
PEER DISCOVERY	19	PSEUDO-ALÉATOIRE	40
PEERS.DAT	20	PULL REQUEST	41
PERCOLATION	21	PYTHON	42

Q

Titre	Page	Titre	Page
QUBIT	1		

R

Titre	Page	Titre	Page
RACINE DE MERKLE	1	RESYNCHRONISATION	11
RBF (REPLACE-BY-FEE)	2	RÉUTILISATION D'ADRESSE	12
RÉCOMPENSE DE BLOC	3	RICOCHET	13
RÉCURSIF (COVENANT)	4	RIPEMD160	14
REDEEMSCRIPT	5	RPC (REMOTE PROCEDURE CALL)	15
RÈGLES DE CONSENSUS	6	RPOW (REUSABLE PROOFS OF WORK)	16
RÈGLES DE STANDARDISATION	7	RSMPPS (RECENT SHARED MAXIMUM PAY PER SHARE)	17
RÉORGANISATION	8	RUST	18
RÉSEAU BITCOIN	9	RUST-LIGHTNING	19
RÉSISTANCE AU PARTITIONNEMENT	10		

S

Titre	Page	Titre	Page
SAMOURAI WALLET	1	SIGHASH_ANYPREVOUT	29
SATOSHI (SAT)	2	SIGHASH_ANYPREVOUTANYSCRIPT	30
SATOSHI NAKAMOTO	3	SIGHASH FLAG	31
SCALA	4	SIGHASH_NONE (0X02)	32
SCHNORR (PROTOCOLE)	5	[SIGHASH_NONE	SIGHASH_ANYONECA (0X82)](#sighash_none-sighash_anyonecanpay-0x82)
SCORE (SCORE BASED METHOD)	6	SIGHASH_SINGLE (0X03)	34
SCRIPT	7	[SIGHASH_SINGLE	SIGHASH_ANYONECA (0X83)](#sighash_single sighash_anyonecanpay-0x83)
SCRIPTLESS SCRIPTS	8	SIGNATURE NUMÉRIQUE	36
SCRIPTPUBKEY	9	SIGNET	37
SCRIPTSIG	10	SIGOPS (SIGNATURE OPERATIONS)	38
SCRIPTWITNESS	11	SLIP (SATOSHI LABS IMPROVEMENT PROPOSALS)	39
SDK (SOFTWARE DEVELOPMENT KIT)	12	SMPPS (SHARED MAXIMUM PAY PER SHARE)	40
SECP256K1	13	SOFT FORK	41
SEED NODES	14	SOMME DE CONTRÔLE (CHECKSUM)	42
SEGWIT	15	SOROBAN	43
SEGWIT2X	16	SORTIE (OUTPUT)	44

Titre	Page	Titre	Page
SEGWIT V0	17	SORTIE NON RENTABLE	45
SEGWIT V1	18	SPEEDY TRIAL	46
SÉLECTION DES PIÈCES	19	SPOF (POINT DE DÉFAILLANCE UNIQUE)	47
SELF-CUSTODY	20	SPREAD (WST)	48
SELFISH MINING	21	STABLECOIN	49
SHA256	22	STALE BLOCK	50
SHA512	23	STONEWALL X2	51
SHAREDCOIN	24	STRATUM	52
SHOR (ALGORITHME)	25	STRATUM V2	53
SIDECHAIN	26	SUBVENTION DE BLOC	54
SIGHASH_ALL (0X01)	27	SURCOUCHE (LAYER)	55
[SIGHASH_ALL SIGHASH_ALL_ANYONECANPAY (0X81)](#sighash_all-sighash_anyonecanpay-0x81)	28	SYNCHRONISATION INITIALE D'UN NŒUD (IBD)	

T

Titre	Page	Titre	Page
TAPROOT	1	TIMELOCK	10
TAPROOT ASSETS PROTOCOL	2	TPRV	11
TAPSCRIPT	3	TPUB	12
TARO	4	TRANSACTION (TX)	13
TAUX DE HACHAGE	5	TRANSACTION COINBASE	14
TCP (TRANSMISSION CONTROL PROTOCOL)	6	TRANSACTION D'ENGAGEMENT	15
TÉMOIN DE TRANSACTION	7	TUMBLEBIT	16
TESTNET	8	TWO-WAY PEG (2WP)	17
TIDES (TRANSPARENT INDEX OF DISTINCT EXTENDED SHARES)	9	TXID (TRANSACTION IDENTIFIER)	18

U

Titre	Page	Titre	Page
UASF (USER-ACTIVATED SOFT FORK)	1	URI (UNIFORM RESOURCE IDENTIFIER)	5
UDP (USER DATAGRAM PROTOCOL)	2	UTXO	6
UPRV	3	UTXO SET	7
UPUB	4		

V

Titre	Page	Titre	Page
VANITY (ADDRESS)	1	VPRV	3
VANITYGEN	2	VPUB	4

W

Titre	Page	Titre	Page
WABISABI	1	WATCHMEN	7
WALLET	2	WATCH-ONLY WALLET	8
WALLET.DAT	3	WHIRLPOOL	9
WALLETS/DB.LOG	4	WHIRLPOOL STAT TOOL	10
WALLET IMPORT FORMAT (WIF)	5	WITNESSSCRIPT	11
WASABI WALLET	6	WTXID	12

X

Titre	Page	Titre	Page
XOR	1	XPUB	3
XPRV	2		

Y

Titre	Page	Titre	Page
YPRV	1	YPUB	2

Z

Titre	Page	Titre	Page
ZEROCONF	1	ZKP (ZERO-KNOWLEDGE PROOF)	4
ZEROLINK	2	ZPRV	5
ZEROSYNC	3	ZPUB	6

A

ACINQ

Entreprise basée en France spécialisée dans le développement de solutions pour le Lightning Network. Fondée en 2014, ACINQ est très actif dans le développement du protocole. Ils sont notamment connus pour avoir créé Eclair, une des 3 implémentations majeures du Lightning Network (avec Core-Lightning et LND). Ils sont également à l'initiative du portefeuille Phoenix.

ADAPTOR SIGNATURE

Méthode cryptographique permettant de combiner une vraie signature avec une signature supplémentaire (appelée « adaptor signature ») pour révéler une donnée secrète. Cette méthode fonctionne telle que la connaissance de deux éléments parmi la signature valide, l'adaptor signature et le secret permet de déduire le troisième manquant. Une des propriétés intéressantes de cette méthode est que si nous connaissons l'adaptor signature de notre pair et le point spécifique sur la courbe elliptique lié au secret utilisé pour calculer cette adaptor signature, nous pouvons alors dériver notre propre adaptor signature qui correspondra avec le même secret, et ce, sans jamais avoir accédé directement au secret lui-même. Dans un échange entre deux parties prenantes ne se faisant pas confiance, cette technique permet un dévoilement simultané de deux informations sensibles entre les participants. Ce processus élimine la nécessité de confiance lors de transactions instantanées telles qu'un Coin Swap ou un Atomic Swap. Prenons un exemple pour bien comprendre. Alice et Bob souhaitent s'envoyer 1 BTC chacun, mais ils ne se font pas confiance. Ils vont donc utiliser des adaptors signatures pour annihiler le besoin de confiance envers l'autre partie dans cet échange (c'est donc un échange « atomique »). Ils procèdent comme ceci :

- Alice initie cet échange atomique. Elle crée une transaction m_A qui envoie 1 BTC vers Bob. Elle crée une signature s_A qui permet de valider cette transaction grâce à sa clé privée p_A ($P_A = p_A \cdot G$), et en utilisant un nonce n_A et un secret t ($N_A = n_A \cdot G$ et $T = t \cdot G$) :

$$s_A = n_A + t + H(N_A + T \parallel P_A \parallel m_A) \cdot p_A$$

- Alice calcule l'adaptor signature s'_A à partir du secret t et de sa vraie signature s_A :

$$s'_A = s_A - t$$

- Alice envoie à Bob son adaptor signature sA', sa transaction non signée m_A, le point correspondant au secret T et le point correspondant au nonce N_A. Nous appelons ces informations un « adaptor ». Notons qu'avec simplement ces informations, Bob n'est pas en capacité de récupérer le BTC d'Alice.

- En revanche, Bob peut vérifier qu'Alice n'est pas en train de l'entourlouper. Pour ce faire, il vérifie que l'adaptor signature d'Alice s'_A correspond bien à la transaction promise m_A. Si l'équation suivante est juste, alors il est persuadé que l'adaptor signature d'Alice est valide :

$$s'_A \cdot G = N_A + H(N_A + T \parallel P_A \parallel m_A) \cdot P_A$$

- Cette vérification donne à Bob des garanties de la part d'Alice, de telle sorte qu'il peut continuer le processus d'échange atomique sereinement. Il va alors créer à son tour sa propre transaction m_B envoyant 1 BTC à Alice et sa propre adaptor signature s'_B qui sera liée avec le même secret t que seule Alice connait pour le moment (Bob n'a pas connaissance de cette valeur t, mais uniquement de son point correspondant T qu'Alice lui a fourni) :

$$s'_B = n_B + H(N_B + T \parallel P_B \parallel m_B) \cdot p_B$$

- Bob envoie à Alice son adaptor signature s'_B, sa transaction non signée m_B, le point correspondant au secret T et le point correspondant au nonce N_B. Alice peut désormais combiner l'adaptor signature de Bob s'_B avec le secret t, dont elle seule a connaissance, afin de calculer une signature valide s_B pour la transaction m_B qui lui envoie le BTC de Bob :

$$s_B = s'_B + t$$
$$(s'_B + t) \cdot G = N_B + T + H(N_B + T \parallel P_B \parallel m_B) \cdot P_B$$

- Alice diffuse cette transaction m_B signée sur la blockchain Bitcoin afin de récupérer le BTC que Bob lui a promis. Bob prend connaissance de cette transaction sur la blockchain. Il est donc en capacité d'en extraire la signature $s_B = s'_B + t$. À partir de cette information, Bob peut isoler le fameux secret t dont il avait besoin :

$$t = (s'_B + t) - s'_B = s_B - s'_B$$

- Or, ce secret t était la seule information manquante à Bob afin de produire la signature valide s_A, à partir de l'adaptor signature d'Alice s'_A, qui lui permettra de valider la transaction m_A qui envoie un BTC depuis Alice vers Bob. Il calcule alors s_A et diffuse à son tour la transaction m_A :

$$s_A = s'_A + t$$

$$(s'_A + t) \cdot G = N_A + T + H(N_A + T \parallel P_A \parallel m_A) \cdot P_A$$

ADDR

Message réseau anciennement utilisé sur Bitcoin pour communiquer les adresses des nœuds acceptant des connexions entrantes. Cet ancien format, se limitant à 128 bits par adresse, était seulement adapté aux adresses IPv6, IPv4 et aux adresses Tor de version 2. Face à l'arrivée de nouveaux protocoles comme Tor V3 et la nécessité de disposer d'une meilleure évolutivité pour de futur protocoles réseau, le format `addr` a été supplanté par `addrv2`, introduit dans le BIP155.

ADDR.DAT

Nom de l'ancien fichier utilisé dans Bitcoin Core pour stocker des informations sur les pairs (c'est-à-dire, les nœuds) du réseau avec lesquels le nœud de l'utilisateur a interagi ou peut potentiellement interagir. Ce fichier a été remplacé par le fichier peers.dat depuis la version 0.7.0.

ADDRV2

Évolution proposée avec le BIP155 du message `addr` sur le réseau de Bitcoin. Le message `addr` servait à diffuser les adresses de nœuds acceptant des connexions entrantes, mais il était limité à des adresses de 128 bits. Cette taille était adéquate pour les adresses IPv6, IPv4, et Tor V2, mais insuffisante pour d'autres protocoles. La version mise à jour `addrv2` est conçue pour supporter des adresses plus longues, notamment les services cachés Tor v3 de 256 bits, ainsi que d'autres protocoles réseau tels que I2P ou de futurs protocoles.

ADRESSE DE RÉCEPTION

Information utilisée pour recevoir des bitcoins. Une adresse est construite en hachant une clé publique, à l'aide de `SHA256` et de `RIMPEMD160`, et en ajoutant des métadonnées à ce condensat. Les clés publiques utilisées pour construire une adresse de réception font partie du portefeuille de l'utilisateur et sont donc dérivées depuis sa graine. Les adresses SegWit sont composées des informations suivantes :

- Un HRP pour désigner « bitcoin » : `bc` ;
- Un séparateur : `1` ;
- La version de SegWit utilisée : `q` ou `p` ;
- La charge utile : le condensat de la clé publique ;
- La somme de contrôle : un code BCH.

Une adresse de réception peut être représentée sous la forme d'une chaîne de caractères alphanumériques ou sous la forme d'un QR code. Chaque adresse peut être utilisée plusieurs fois, mais c'est une pratique très déconseillée. En effet, dans le but de maintenir un certain niveau de confidentialité, il est conseillé de n'utiliser chaque adresse Bitcoin qu'une seule fois. Il faut en générer une nouvelle pour tout paiement entrant vers son portefeuille. Une adresse est encodée en `Bech32` pour les adresses SegWit V0, en `Bech32m` pour les adresses SegWit V1, et en `Base58check` pour les adresses Legacy. D'un point de vue technique, une adresse ne permet pas réellement de recevoir des bitcoins, mais plutôt de bloquer des bitcoins à l'aide d'un script, en mettant des contraintes sur leur dépense.

AJUSTEMENT DE DIFFICULTÉ

L'ajustement de la difficulté est un processus périodique qui redéfinit la cible de difficulté pour le mécanisme de la preuve de travail (le minage) sur Bitcoin. Cet évènement intervient tous les 2016 blocs (environ toutes les deux semaines). Il vient augmenter ou baisser le facteur de difficulté (également nommé la cible de difficulté), en fonction de la rapidité à laquelle les 2016 derniers blocs ont été trouvés. L'ajustement vise à conserver un taux de production de blocs stable et prévisible, à une fréquence d'un bloc toutes les 10 minutes, malgré les variations de la puissance de calcul déployée par les mineurs. La modification de la difficulté lors de l'ajustement est limitée à un facteur 4. Le calcul qu'effectuent les nœuds pour calculer la nouvelle cible est le suivant : $N = A \cdot \left(\frac{T}{1{,}209{,}600}\right)$
Où :

- N : La nouvelle cible ;
- A : L'ancienne cible des 2016 derniers blocs ;
- T : Le temps total réel des 2016 derniers blocs en secondes ;
- $1,209,600$: Le temps cible en secondes pour produire 2016 blocs avec un intervalle de 10 minutes entre chacun.

  En français, on parle parfois également de « reciblage » pour évoquer l'ajustement. En anglais, on parle de « Difficulty Adjustment ».

ALGORITHME

Suite finie et non ambiguë d'instructions permettant de réaliser une tâche. Dans le cadre de l'informatique, il s'agit d'un processus écrit dans un langage de programmation qui indique à un ordinateur comment effectuer une mission.

ANALYSE DE CHAINE

Pratique qui regroupe toutes les méthodes permettant de tracer les flux de bitcoins sur la blockchain. De façon générale, l'analyse de chaîne s'appuie sur l'observation de caractéristiques sur des échantillons de transactions antérieures. Elle consiste ensuite à repérer ces mêmes caractéristiques sur une transaction que l'on souhaite analyser, et à en déduire des interprétations vraisemblables. Cette méthode de résolution de problème à partir d'une approche pratique, pour trouver une solution suffisamment bonne, c'est ce que l'on appelle une heuristique. Pour vulgariser, l'analyse de chaîne se fait en deux grandes étapes :

- Le repérage de caractéristiques connues ;
- La déduction d'hypothèses.

Un des objectifs de l'analyse de chaîne consiste à regrouper diverses activités sur Bitcoin en vue de déterminer l'unicité de l'utilisateur les ayant effectuées. Par la suite, il sera possible de tenter de rattacher ce faisceau d'activités à une identité réelle grâce à un point d'entrée. Il est primordial de comprendre que l'analyse de chaîne n'est pas une science exacte. Elle repose sur des heuristiques dérivées d'observations antérieures ou d'interprétations logiques. Ces règles permettent d'obtenir des résultats assez fiables, mais jamais d'une précision absolue. En d'autres termes, l'analyse de chaîne implique toujours une dimension de vraisemblabilité dans les conclusions émises. On pourra estimer avec plus ou moins de certitude que deux adresses appartiennent à une même entité, mais une certitude totale sera toujours hors de portée. Tout l'objectif de l'analyse de chaîne réside précisément dans l'agrégation de diverses heuristiques en vue de minimiser le risque d'erreur. Il s'agit en quelque sorte d'une accumulation de preuves qui nous permet de nous approcher davantage de la réalité. Ces fameuses heuristiques peuvent être regroupées en différentes catégories :

- Les patterns de transaction (ou modèles de transaction) ;
- Les heuristiques internes à la transaction ;
- Les heuristiques externes à la transaction.

Notons que les deux premières heuristiques sur Bitcoin ont été formulées par Satoshi Nakamoto lui-même. Il les expose dans la partie 10 du White Paper (livre blanc). Il est intéressant d'observer que ces deux heuristiques conservent toujours une prééminence dans l'analyse de chaîne aujourd'hui. Ce sont :

- la CIOH (Common Input Ownership Heuristic) ;
- et la réutilisation d'adresse.

ANCESTOR MINING

Autre nom parfois donné à CPFP (Child-Pay-For-Parent). Le minage des ancêtres est le principe selon lequel un mineur ne choisit pas une transaction uniquement sur la base de ses propres frais de transaction, mais prend aussi en compte les frais des transactions ascendantes ou descendantes.

*Pour plus d'informations, voir la définition de **CPFP (CHILD PAY FOR PARENT)**.*

ANCHOR OUTPUTS

Proposition qui vise à améliorer la gestion des frais de transaction dans le cadre des canaux Lightning. À chaque changement d'état dans un canal Lightning, les parties prenantes créent et signent une nouvelle transaction d'engagement, reflétant la nouvelle répartition des fonds au sein du canal. Le problème de ce mécanisme réside dans la détermination des frais de transaction au moment de sa

création. En effet, les frais de transaction sur le réseau Bitcoin sont sujets à de fortes fluctuations, tant à la hausse qu'à la baisse. Si les frais fixés pour la dernière transaction d'engagement sont insuffisants au moment de la fermeture unilatérale du canal, non seulement la transaction prendra un temps considérable à se confirmer, mais les mécanismes de verrouillage temporel (timelocks) pourraient également permettre un vol des fonds. Les anchor outputs permettent de réserver une petite partie des fonds dans une transaction d'engagement pour couvrir les frais futurs. En cas de congestion du réseau et d'augmentation des frais, les anchor outputs permettent de modifier les frais de transaction après la création de la transaction d'engagement, garantissant ainsi une fermeture suffisamment rapide du canal Lightning.

ANCHORS.DAT

Fichier utilisé dans le client Bitcoin Core pour stocker les adresses IP des nœuds sortants auxquels un client était connecté avant d'être éteint. Anchors.dat est donc créé à chaque fois que le nœud est arrêté et supprimé lorsqu'il est relancé. Les nœuds dont les adresses IP sont contenues dans ce fichier sont utilisés pour aider à établir rapidement des connexions lors du redémarrage du client.

ANCRAGE BILATÉRAL

Mécanisme qui permet d'établir une connexion entre le système principal de Bitcoin et une sidechain (ou une drivechain), c'est-à-dire une chaîne latérale. L'ancrage bilatéral assure une corrélation de valeur fixe entre les bitcoins sur la blockchain principale et les actifs correspondants sur la sidechain, permettant ainsi de déplacer des bitcoins entre les deux chaînes. Pour ce faire, les bitcoins sont temporairement verrouillés sur la blockchain principale et un montant équivalent d'actifs est émis sur la sidechain. Cela permet de profiter des avantages spécifiques de la sidechain, comme des transactions plus rapides ou des fonctionnalités de confidentialité améliorées, tout en maintenant la valeur des bitcoins utilisés. Lorsque les utilisateurs souhaitent revenir à la blockchain Bitcoin, le processus s'inverse : les actifs sur la sidechain sont détruits et les bitcoins correspondants sont déverrouillés. Il existe de nombreux mécanismes d'ancrages bilatéraux différents qui peuvent reposer sur :

- Un tiers de confiance unique ;
- Une fédération d'entités ;
- Les mineurs de la chaîne principale (drivechain).

  En anglais, on parle d'un « two-way peg » ou « 2WP ».

ANONSETS (ANONYMITY SETS)

Les anonsets servent d'indicateurs pour évaluer le degré de confidentialité d'un UTXO particulier. Plus spécifiquement, ils mesurent le nombre d'UTXOs indistinguables au sein de l'ensemble qui inclut la pièce en étudiée. Puisqu'il faut disposer d'un groupe d'UTXOs identiques, les anonsets sont généralement calculés au sein d'un cycle de CoinJoins. Il permettent, le cas échéant, de juger de la qualité des CoinJoins. Un anonset de grande taille signifie un niveau d'anonymat accru, car il devient difficile de distinguer un UTXO spécifique au sein de l'ensemble. Deux types d'anonsets existent :

- L'ensemble d'anonymat prospectif ;
- L'ensemble d'anonymat rétrospectif.

Le premier indique la taille du groupe parmi lequel se cache l'UTXO étudié en sortie, sachant l'UTXO en entrée. Cet indicateur permet de mesurer la résistance de la confidentialité de la pièce face à une

analyse passé vers présent (entrée vers sortie). En anglais, le nom de cet indicateur est « forward anonset », ou « forward-looking metrics ». Le second indique le nombre de sources possibles pour une pièce donnée, sachant l'UTXO en sortie. Cet indicateur permet de mesurer la résistance de la confidentialité de la pièce face à une analyse présent vers passé (sortie vers entrée). En anglais, le nom de cet indicateur est « backward anonset », ou « backward-looking metrics ».

> *En français, il est globalement admis d'utiliser le terme « anonset ». On pourrait toutefois le traduire par « ensemble d'anonymat » ou « potentiel d'anonymat ». En anglais et en français, on parle également parfois de « score » pour évoquer les anonsets (score prospectif et score rétrospectif). Pour plus d'informations, voir la définition* ***COINJOIN****.*

ANYPREVOUT (APO)

Nom donné au BIP118 qui propose d'ajouter deux nouveaux SigHash Flag modificateurs, nommés `SIGHASH_ANYPREVOUT` et `SIGHASH_ANYPREVOUTANYSCRIPT`. Le terme « *AnyPrevOut* » provient de la contraction de « *Any Previous Output* » que l'on pourrait traduire en français par « toute sortie précédente ». Pour plus d'informations, voir les définitions correspondantes.

AOPP

Sigle de « *Address Ownership Proof Protocol* ». C'est un protocole controversé, conçu pour prouver automatiquement la propriété d'adresses Bitcoin. Ce mécanisme permet aux utilisateurs de démontrer qu'ils contrôlent une adresse spécifique, directement à travers leur logiciel de portefeuille compatible. Initialement, l'AOPP a été créé pour simplifier la vérification de possession d'adresses, une exigence légale pour les clients désirant transférer leurs bitcoins hors des plateformes d'échange dans certaines juridictions, telles que la Suisse. Néanmoins, ce protocole a été l'objet de critiques importantes au sein de la communauté Bitcoin, car il pourrait établir un précédent où les utilisateurs devraient demander l'autorisation pour exercer leur droit de possession sur leurs propres fonds (self-custody). Face à ces critiques, de nombreux logiciels de portefeuille ont choisi de ne pas adopter ce protocole.

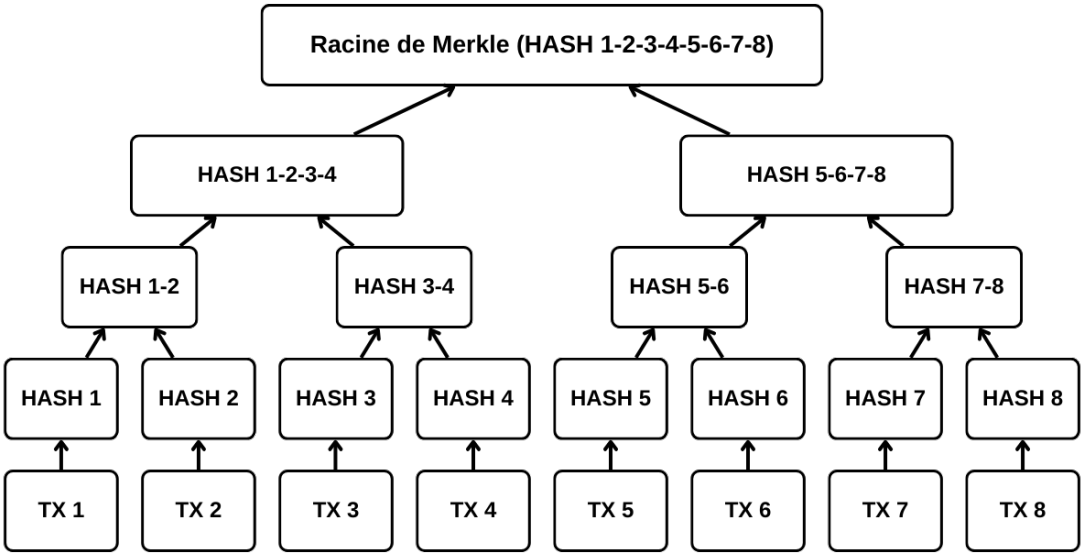
API

Sigle de « *Application Programming Interface* ». Dans le contexte général de l'informatique, une API est un ensemble de règles et de spécifications que les logiciels peuvent suivre pour communiquer entre eux. Elles permettent aux développeurs d'accéder à des fonctionnalités ou à des données d'une application, d'un système d'exploitation ou d'un autre service pour leur propre logiciel.

> *En français, on peut le traduire par « interface de programmation d'applications » ou directement « interface de programmation ».*

ARBRE DE MERKLE

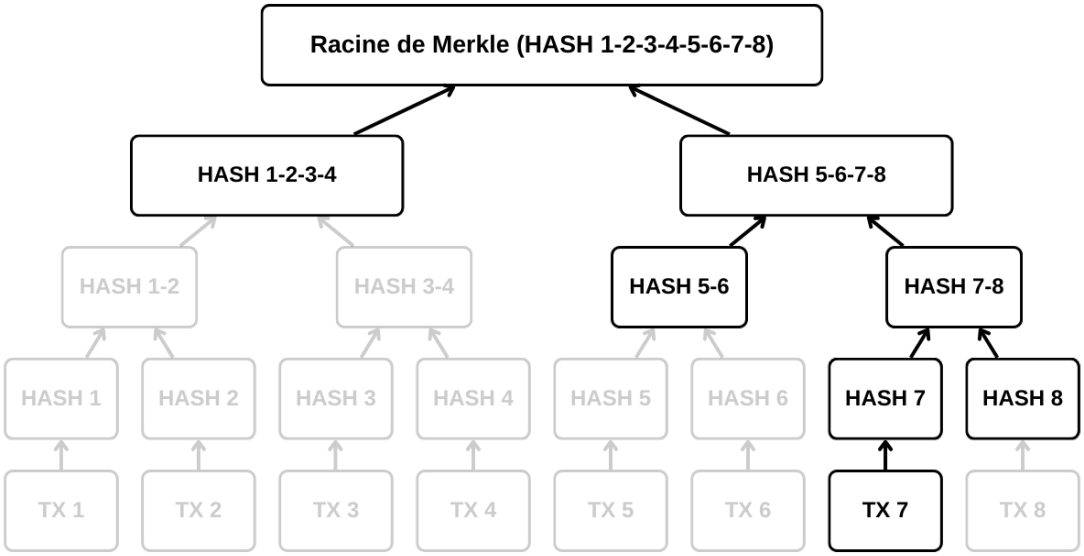
Un Arbre de Merkle est un accumulateur cryptographique. C'est une méthode pour justifier l'appartenance d'une information donnée à un ensemble plus grand. C'est une structure de données qui facilite la vérification d'informations dans un format compact. Dans le système Bitcoin, les arbres de Merkle sont utilisés pour regrouper et condenser les transactions d'un bloc en un unique hachage, appelé la racine de Merkle (ou « Top Hash »). Chaque transaction est hachée, puis les hachages adjacents sont hachés ensemble de façon hiérarchique jusqu'à ce que la racine de Merkle soit obtenue.



Cette structure permet de vérifier rapidement si une transaction spécifique est incluse dans un bloc donné sans avoir à analyser l'ensemble des transactions. Par exemple, si je dispose seulement de la racine de Merkle et que je souhaite vérifier que la `TX 7` fait bien partie de l'arbre, j'aurai uniquement besoin des preuves suivantes :

- `TX 7` ;
- `HASH 8` ;
- `HASH 5-6` ;
- `HASH 1-2-3-4` .

Grâce à ces quelques informations, je suis en capacité de calculer les nœuds intermédiaires jusqu'à la racine de Merkle.



Les arbres de Merkle sont notamment utilisés pour les nœuds légers, dits « SPV Node », qui ne conservent que les entêtes de blocs, mais pas les transactions. On retrouve également cette structure dans le protocole UTREEXO, une structure permettant de condenser l'UTXO set des nœuds, et dans le MAST Taproot.

> *L'arbre de Merkle porte le nom de Ralph Merkle, un cryptographe pionnier qui a conçu cette structure en 1979. Un arbre de Merkle peut également être nommé « arbre de hachage ». En anglais, on dit « Merkle Tree » ou « Hash Tree ».*

ARK

Nouveau protocole de seconde couche dévoilé par Burak en mai 2023. Comme le Lightning Network, Ark est un système se déployant par-dessus la chaîne principale de Bitcoin. Il permettrait de faire des paiements en bitcoins en dehors de la chaîne de manière rapide, anonyme et à bas frais. Par rapport à Lightning, Ark ne nécessite pas d'avoir des liquidités entrantes pour recevoir des paiements, ce qui permet d'améliorer considérablement l'expérience utilisateur. De plus, il procure une confidentialité se rapprochant des transactions coinjoins, alors que Lightning est un très mauvais modèle pour préserver sa vie privée. Enfin, Ark pourrait également être non interactif si des covenants sont ajoutés à Bitcoin. Burak critique souvent la capacité de Lightning à passer à l'échelle en raison de sa dépendance à la chaîne principale et suggère qu'Ark pourrait théoriquement intégrer toute la population mondiale en self-custody. Même si Ark peut être vu comme un protocole concurrent au Lightning Network, les deux peuvent en réalité coexister. Ils pourraient même être plutôt complémentaires. Notons toutefois que pour le moment, Ark n'est qu'une simple idée. Burak n'a pas encore dévoilé le code de son invention.

ASIC

Un ASIC est un composant électronique conçu pour exécuter une fonction spécifique avec une efficacité optimale. Dans le contexte du minage de Bitcoin, les ASIC sont des circuits intégrés spécialisés qui effectuent des opérations de hachage à haute vitesse et faible consommation d'énergie. Ils

sont spécialisés dans l'exécution de la fonction de hachage `SHA256` utilisée dans le mécanisme de la preuve de travail. L'ASIC est initialement le nom de la puce. Par extension, l'acronyme « ASIC » vise souvent à désigner également la machine qui héberge cette puce. Ainsi, les ordinateurs spécialisés dans le minage de Bitcoin sont parfois appelés des « ASIC », ou bien des « mineurs ». Les ASIC ont progressivement remplacé les autres méthodes de minage, telles que l'utilisation de processeurs (CPU) et de cartes graphiques (GPU), en raison de leur efficacité énergétique supérieure et de leur taux de hachage bien plus élevé.

> *L'acronyme « ASIC » désigne en anglais « Application-Specific Integrated Circuit ». En français, ce terme peut être traduit par « Circuit intégré spécifique à une application ».*

ASMAP

Outil inventé par Gleb Naumenko et utilisé par Bitcoin Core pour améliorer la sécurité et la topologie du réseau Bitcoin en diversifiant les connexions entre les nœuds. Il s'agit d'une carte d'adressage IP vers les numéros de systèmes autonomes (ASN), permettant une meilleure répartition des connexions sortantes en fonction de l'ASN plutôt que des préfixes IP. Cela aide à prévenir les attaques Eclipse (notamment l'attaque Erebus) en rendant plus difficile pour un attaquant de simuler plusieurs nœuds.

ASSUME UTXO

Paramètre de configuration dans le client majoritaire Bitcoin Core qui permet à un nœud qui vient d'être initialisé (mais qui n'a pas encore fait l'IBD) de reporter la vérification des transactions et de l'UTXO set avant un snapshot donné. Le concept repose sur l'utilisation d'un UTXO set (liste de tous les UTXOs existants à un moment donné) fourni par Core et présumé exact, ce qui permet au nœud d'être synchronisé très rapidement sur la chaîne avec le plus de preuve de travail accumulée. Puisque le nœud saute la longue étape de l'IBD, il est très rapidement fonctionnel pour son utilisateur. Assume UTXO divise la synchronisation (IBD) en deux parties :

- Tout d'abord, le nœud réalise le Header First Sync (vérification des en-têtes seulement) et il considère comme valide l'UTXO set qui lui est fourni par Core ;
- Puis, une fois qu'il est fonctionnel, le nœud va vérifier l'historique complet des blocs en arrière-plan, en actualisant un nouvel UTXO set qu'il aura vérifié lui-même. Si ce dernier ne correspond pas à l'UTXO set fourni par Core, il fournira un message d'erreur.

Assume UTXO permet donc d'accélérer la préparation d'un nouveau nœud Bitcoin en reportant le processus de vérification des transactions et de l'UTXO set grâce à un snapshot actualisé fourni dans Core.

ASSUME VALID

Paramètre de configuration dans le client majoritaire Bitcoin Core qui permet à un nœud qui vient d'être initialisé (mais qui n'a pas encore fait l'IBD) de sauter la vérification des signatures pour toutes les transactions inclues dans les blocs antérieur à un certain bloc donné. Ce fameux bloc est défini par l'empreinte de son en-tête, c'est-à-dire son hash. Le bloc choisi est renouvelé lors de chaque nouvelle version de Bitcoin Core. À son initialisation, si le nœud a activé ce paramètre, il va donc vérifier la chaîne d'en-têtes de blocs pour trouver la branche avec le plus de preuve de travail accumulée. Si le nœud détecte le hash fourni par Core dans la branche qu'il a retenue, il omettra la vérification des signatures pour les blocs antérieurs. Dans le cas contraire, le nœud procédera à une synchronisation traditionnelle (IBD) pour tout vérifier par lui-même. L'objectif d'Assume Valid est d'accélérer le processus de synchronisation initiale d'un nœud sans compromettre la sécurité, en supposant que la majorité du réseau a déjà validé ces transactions dans le passé. Le seul vrai

compromis pour le nœud est qu'en cas de vol antérieur de bitcoins, il ne sera pas averti. Cependant, il peut toujours s'assurer de l'exactitude de la quantité de bitcoins émis. Les nœuds poursuivent la vérification des signatures de transactions postérieures au bloc Assume Valid. Cette approche repose sur l'hypothèse que si une transaction est acceptée par le réseau depuis assez longtemps sans contestation, il est improbable qu'elle soit frauduleuse.

ATH (ALL-TIME HIGH)

Désigne le niveau le plus élevé jamais atteint par l'élément étudié. Souvent, l'ATH désigne le plus haut niveau de prix du bitcoin en comparaison avec une monnaie étatique sur une période donnée.

ATLC

Sigle de « *Anchor Timelock Contracts* ». C'est un paiements conditionnels utilisés dans le cadre du protocole Ark pour fournir un calendrier de paiement atomique à un hub, grâce à des connecteurs permettant de former ce que l'on appelle un « txlock ». L'objectif d'un ATLC est sensiblement le même que celui d'un HTLC sur Lightning.

*Pour plus d'informations, voir la définition de **ARK**.*

ATOMIC SWAP

Technologie permettant un échange de cryptomonnaies directement entre deux parties sans besoin de confiance et sans nécessiter d'intermédiaire. Ces échanges sont dits « atomiques » car ils ne peuvent donner que deux résultats :

- Soit l'échange réussi et les deux participants se sont effectivement échangé leurs cryptomonnaies ;
- Soit l'échange échoue et les deux participants repartent avec leurs cryptomonnaies de départ.

Les Atomic Swaps peuvent s'effectuer soit avec une même cryptomonnaie, dans ce cas on parle également de « Coin Swap », soit entre des cryptomonnaies différentes. Historiquement, ils s'appuyaient sur des « *Hash Time-Locked Contracts* » (HTLC), un système de verrouillage temporel qui garantie la complétude ou l'annulation totale de l'échange, préservant ainsi l'intégrité des fonds des parties impliquées. Cette méthode exigeait des protocoles capables de gérer à la fois les scripts et les timelocks. Toutefois, ces dernières années, la tendance s'est orientée vers l'utilisation des Adaptor Signatures. Cette seconde approche présente l'avantage de se passer de scripts, réduisant ainsi les coûts opérationnels. Son autre atout majeur réside dans le fait qu'elle n'exige pas l'emploi d'un hachage identique pour les deux volets de la transaction, évitant ainsi de révéler un lien entre elles.

ATTAQUE DES 51%

Scénario hypothétique sur le système Bitcoin où un acteur malveillant contrôle plus de 50% de la puissance de calcul totale du minage (hashrate). Avec une telle dominance, l'attaquant peut manipuler le processus de consensus, permettant des actions malveillantes telles que la double dépense, où les mêmes bitcoins sont dépensés une première fois sur une chaîne finalement rendue désuète, puis une seconde fois sur la chaîne valide. Une autre finalité d'une attaque des 51% est la censure des transactions. Cependant, réaliser une attaque des 51% nécessite des ressources financières, humaines, énergétiques et techniques considérables, et rend l'acteur malveillant susceptible d'être découvert avant que l'attaque n'ait lieu. Bien que théoriquement possible, une attaque des 51% sur

Bitcoin est considérée comme très peu probable en raison de la décentralisation du minage et de la grande puissance de calcul actuellement déployée.

Cette attaque est également nommée « Attaque Goldfinger ».

B

BANLIST.DAT

Nom de l'ancien fichier utilisé par le logiciel Bitcoin Core pour enregistrer les adresses IP des nœuds qui ont été bannis par l'utilisateur. Depuis la version 22.0, on utilise le fichier banlist.json à la place.

BANLIST.JSON

Nom du fichier utilisé par le logiciel Bitcoin Core pour enregistrer les adresses IP des nœuds qui ont été bannis par l'utilisateur. Ce fichier contient donc une liste des nœuds bannis avec lesquels le nœud ne se connectera pas. Cette fonctionnalité permet d'empêcher les interactions avec des nœuds potentiellement nuisibles ou malveillants.

BARE-MULTISIG

Script de type P2MS.

*Pour plus d'informations, voir la définition de **P2MS**.*

BASE (ARITHMÉTIQUE)

Une base est un système de numération positionnel qui utilise un nombre fixe de caractères pour représenter tous les nombres possibles. La base détermine le nombre de symboles distincts disponibles pour représenter les chiffres dans ce système. Par exemple, le système le plus connu dans nos vies quotidiennes est la base 10, également appelée système décimal. Elle utilise dix symboles distincts `(0, 1, 2, 3, 4, 5, 6, 7, 8, 9)` pour représenter tous les nombres. D'autres systèmes de numération couramment utilisés dans les domaines informatique et mathématique incluent le système binaire (base 2), avec deux symboles `(0, 1)`, et le système hexadécimal (base 16), avec seize symboles `(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F)`. Dans le cadre de Bitcoin, vous rencontrerez parfois des encodages en base 58 ou en base 32 adaptée (nommée `Bech32`).

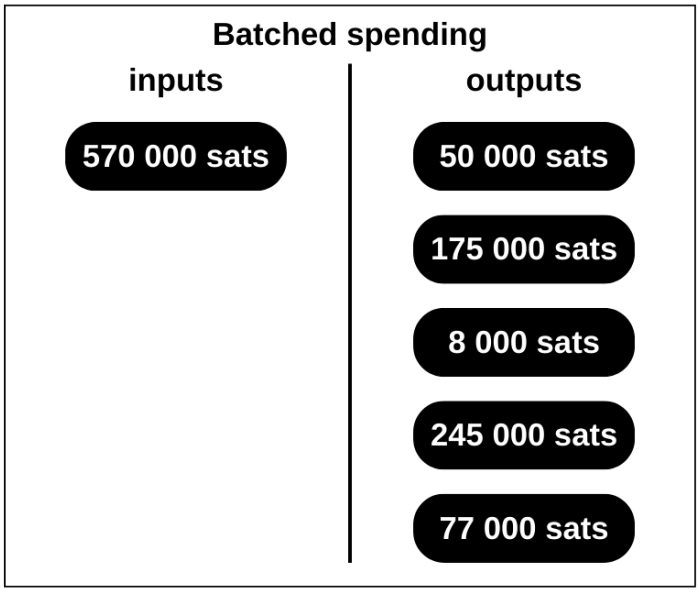
BASE58CHECK

La `Base58Check` est un encodage utilisé dans le système Bitcoin pour représenter les adresses de réception Legacy et certaines autres données, telles que les clés étendues, sous forme de chaînes de caractères lisibles par l'homme. C'est une variante du système `Base58`, une représentation positionnelle de base 58 conçue pour minimiser les erreurs de transcription humaine. Elle utilise un ensemble de 58 caractères alphanumériques, composé des chiffres de `1` à `9`, des lettres majuscules `A` à `Z` (à l'exception des lettres `I` et `O` pour éviter la confusion avec les chiffres `1` et `0`) et des lettres minuscules de `a` à `z` (à l'exception de la lettre `l` pour éviter la confusion avec le chiffre `1`). La `Base58Check` se distingue de la `Base58` par l'ajout d'une somme de contrôle (checksum). Elle est représentée par une version réduite d'un double hachage `SHA256` des données originales (`SHA256d` ou `HASH256`), à la fin des données encodées en `Base58`. Lors de la vérification, la somme de contrôle est recalculée et comparée à celle qui a été ajoutée lors de l'encodage. Si les deux empreintes correspondent, les données sont considérées comme valides, sinon une erreur de corruption ou de transcription est signalée. L'utilisation de la `Base58Check` dans les adresses Bitcoin et les clés privées procure plusieurs avantages. Premièrement, elle permet de réduire les erreurs humaines lors de la transcription et de la lecture en évitant les caractères ambigus. Deuxièmement, elle protège contre les erreurs de saisie en détectant et signalant les erreurs grâce au hachage de vérification. Troisièmement, la représentation compacte des données en `Base58Check` permet de réduire l'espace requis pour stocker et partager les adresses et les clés. Les adresses de réception les plus récentes (post-SegWit) ont abandonné

cet encodage `Base58check` pour des encodages `Bech32` et `Bech32m`, disposant d'une somme de contrôle plus évoluée (code BCH).

BATCHED SPENDING

Technique de dépense employée principalement par les entités ayant un volume élevé de transactions, comme les plateformes d'échange par exemple, pour optimiser et réduire les coûts de transaction. En regroupant plusieurs paiements destinés à différents destinataires en une seule transaction Bitcoin, la dépense groupée permet de consommer moins d'espace ans les blocs, ce qui permet de réduire les frais associés. Le batched spending se distingue par un modèle facilement reconnaissable lors d'une analyse de chaîne. Ce modèle se manifeste par l'utilisation de quelques UTXO en entrée (souvent un seul) et la création de multiples UTXO en sortie. L'interprétation de ce modèle est que nous sommes en présence d'une dépense groupée. Face à ce modèle en analyse de chaîne, on peut en déduire que l'UTXO en entrée provient d'une société avec une grosse activité économique et que les UTXO en sorties vont se disperser. Certains appartiendront à des clients de la société. D'autres iront peut-être vers des sociétés partenaires. Enfin, il y aura certainement un change qui reviendra à la société émettrice.



En français, on peut traduire « batched transaction » par « dépense groupée ».

BDK (BITCOIN DEV KIT)

Kit de développement (SDK) pour les portefeuilles sur Bitcoin. BDK est une collection de bibliothèques et d'outils destinés aux développeurs, permettant de simplifier la création d'applications de portefeuilles Bitcoin. BDK fournit des modules de fonctionnalités essentielles telles que la gestion de portefeuilles, la construction de transactions, la signature de transactions ou encore la gestion des clés. Les développeurs peuvent ensuite s'appuyer sur ces modules pour concevoir leurs propres logiciels. Les composants de BDK sont élaborés dans un souci de légèreté et de modularité, afin de les rendre ajustables à la plupart des possibilités d'utilisation. L'objectif de cet outil est de centraliser le développement de portefeuilles Bitcoin afin de concentrer les efforts.

BDK était auparavant appelé « Magical Bitcoin ».

BECH32 ET BECH32M

`Bech32` et `Bech32m` sont deux formats d'encodage d'adresse pour recevoir des bitcoins. Ils sont établis sur une base 32 légèrement modifiée. Ils embarquent une somme de contrôle établie sur un algorithme de correction d'erreurs appelé BCH (Bose-Chaudhuri-Hocquenghem). Par rapport aux adresses Legacy, encodées en `Base58check`, les adresses `Bech32` et `Bech32m` disposent d'une somme de contrôle plus performante, permettant de détecter et potentiellement de modifier automatiquement les fautes de frappe. Leur format dispose également d'une meilleure lisibilité, avec uniquement des caractères minuscules. Voici la matrice d'addition de ce format depuis la base 10 :

+	0	1	2	3	4	5	6	7
0	q	p	z	r	y	9	x	8
8	g	f	2	t	v	d	w	0
16	s	3	j	n	5	4	k	h
24	c	e	6	m	u	a	7	l

`Bech32` et `Bech32m` sont des formats d'encodage utilisés pour représenter les adresses SegWit. `Bech32` est un format d'encodage d'adresse introduit par la BIP173 en 2017. Il utilise un ensemble de caractères spécifiques, composé de chiffres et de lettres minuscules, pour minimiser les erreurs de frappe et faciliter la lecture. Les adresses `Bech32` commencent généralement par `bc1` pour indiquer qu'elles sont natives de SegWit. Ce format est uniquement utilisé sur les adresses SegWit V0, avec les scripts `P2WPKH` (Pay to Witness Public Key Hash) et `P2WSH` (Pay to Witness Script Hash). Toutefois, il existe une petite faille inattendue propre au format `Bech32`. Chaque fois que le dernier caractère de l'adresse est un `p`, l'ajout ou la suppression d'un nombre quelconque de caractères `q` le précédant immédiatement n'invalide pas la somme de contrôle. Cela n'affecte pas les utilisations existantes des adresses SegWit V0 (BIP173) en raison de leur restriction à deux longueurs définies. Cependant, cela pourrait affecter des utilisations futures de l'encodage `Bech32`. Le format `Bech32m` est simplement un format `Bech32` avec cette erreur rectifiée. Il a été introduit avec le BIP350 en 2020. Les adresses `Bech32m` commencent également par `bc1`, mais elles sont spécifiquement conçues pour être compatibles avec la version SegWit V1 (Taproot) et les versions ultérieures, avec le script `P2TR` (Pay to TapRoot).

BERKELEYDB

Système de gestion de base de données embarquable avec une architecture de stockage clé-valeur. Il a été utilisé dans les premières versions de Bitcoin et a été remplacé par LevelDB en 2012.

BIG-ENDIAN

Format de stockage de données dans les systèmes informatiques où les octets les plus significatifs (les « gros bouts ») sont placés en premier dans l'ordre des adresses. Cela signifie que dans une séquence avec plusieurs octets, l'octet ayant le plus grand poids (par exemple, les chiffres les plus à gauche en hexadécimale) est stocké en premier.

BIP (BITCOIN IMPROVEMENT PROPOSAL)

Une proposition d'amélioration de Bitcoin (BIP) est un processus formel de proposition et de documentation des améliorations et des modifications apportées au protocole Bitcoin et à ses normes. Inspiré

du processus des Python Enhancement Proposals (PEP), le BIP vise à faciliter la communication et la collaboration entre les développeurs, les chercheurs, les utilisateurs et les parties prenantes de l'écosystème Bitcoin. Le processus BIP assure une approche structurée et transparente pour l'évaluation et l'adoption de nouvelles fonctionnalités, optimisations et mises à jour. Chaque BIP est un document détaillé qui décrit précisément les objectifs de l'amélioration proposée, la justification de sa mise en œuvre, les éventuels problèmes de compatibilité, les avantages et les inconvénients. Il décrit également les étapes techniques nécessaires pour réaliser l'amélioration. Les BIP peuvent être rédigés par n'importe qui. Ils doivent cependant être soumis à un examen approfondi et à l'approbation d'autres membres de la communauté Bitcoin.

> *BIP est l'acronyme anglais pour « Bitcoin Improvment Proposal ». En français, on peut le traduire par « Proposition d'amélioration de Bitcoin ». Toutefois, la plupart des textes français utilisent directement l'acronyme « BIP » comme un nom commun, parfois au féminin, parfois au masculin.*

BIP1

Document qui définit le processus d'élaboration et de mise en œuvre des améliorations proposées au protocole Bitcoin. Rédigé par Amir Taaki en août 2011, il établit une méthodologie standard pour proposer et documenter des modifications potentielles. Les propositions de BIP sont alors classées en trois catégories :

- Standards Track : ceux qui concernent les modifications directes du protocole Bitcoin et son interopérabilité ;
- Informational : ceux qui offrent des informations générales mais n'impactent pas directement le protocole ;
- Process : ceux qui introduisent des changements non techniques, comme les procédures et lignes directrices.

Ce cadre systématise le développement de Bitcoin, afin d'assurer une approche coordonnée et transparente de ses évolutions. La méthodologie du BIP1 sera par la suite remplacée par celle du BIP2.

BIP2

Document rédigé par Luke Dashjr en juin 2012 qui établit des critères précis pour l'acceptation des futurs BIP *(Bitcoin Improvement Proposals*). BIP2 introduit le concept de BIP « accepté », « rejeté » ou « retiré », et précise les conditions nécessaires à chaque statut. Un BIP est considéré comme « accepté » s'il reçoit un soutien significatif de la communauté Bitcoin, notamment des développeurs et des utilisateurs. Un BIP est « rejeté » lorsqu'il n'obtient pas suffisamment de soutien ou présente des problèmes insurmontables. Enfin, un BIP peut être « retiré » par son auteur à tout moment. BIP2 redéfinie aussi plus précisément la méthodologie et le format introduits avec le BIP1.

BIP8

Élaboré suite aux débats sur SegWit qui utilisait le BIP9 pour son activation, le BIP8 est une méthode d'activation de soft forks qui incorpore nativement un mécanisme d'UASF (*User-Activated Soft Fork*) automatique. Comme le BIP9, le BIP8 utilise la signalisation des mineurs, mais ajoute le paramètre `LOT` (*Lock-in On Time out*). Si `LOT` est réglé sur `vrai`, à l'expiration de la période de signalisation sans atteindre le seuil requis, un UASF est automatiquement déclenché, forçant l'activation du soft fork. Cette approche contraint les mineurs à être coopératifs ou risquer un UASF imposé par les utilisateurs. De plus, contrairement au BIP9, le BIP8 définit la période de signalisation basée sur la hauteur des

blocs, éliminant les manipulations potentielles via le taux de hachage par les mineurs. Le BIP8 permet également de fixer un seuil de vote variable et introduit un paramètre pour une hauteur de bloc minimale pour l'activation, donnant aux mineurs le temps de se préparer et de signaler leur accord en avance sans forcément être prêts. Lorsqu'un soft fork est activé via le BIP8 avec le paramètre `LOT=vrai`, on utilise ici une méthode très agressive contre les mineurs qui sont immédiatement mis sous la pression d'un éventuel UASF. En effet, cela leur laisse seulement 2 choix :

- Être coopératifs, et ainsi faciliter le processus d'activation ;
- Être non coopératifs, auquel cas les utilisateurs font un UASF automatiquement pour imposer le soft fork.

BIP9

Méthode d'activation de soft forks sur la blockchain Bitcoin, proposée en 2015. Elle introduit un système où les mineurs signalent leur soutien à un soft fork en utilisant un bit spécifique dans le champ de version des blocs. Un soft fork proposé sous BIP9 est activé si 95% des blocs sur une période de 2016 blocs (environ deux semaines, coïncidant avec chaque ajustement de difficulté) signalent leur approbation. Après ce verrouillage, un délai est accordé pour que les mineurs se préparent à la mise à jour avant son activation. En cas d'échec à atteindre le seuil de 95% dans la durée maximale prévue, le soft fork est abandonné. BIP9 permet la signalisation de plusieurs soft forks simultanément mais donne un pouvoir considérable aux mineurs, car si le seuil requis n'est pas atteint, le soft fork est simplement abandonné. Cette méthode était celle initialement utilisée pour SegWit, avant que le BIP148 qui suggère l'utilisation d'un UASF ne vienne rebattre les cartes et forcer le verrouillage via le BIP91.

BIP10

Le BIP10, proposé par Gavin Andresen en 2011, visait à introduire le concept de transactions multi-signatures. L'idée principale de BIP10 était de standardiser la façon de conditionner un paiement à plusieurs signatures (multisig) en introduisant un nouveau type de script. Cette proposition suggérait de permettre aux émetteurs de transactions de spécifier plusieurs clés publiques et une condition selon laquelle un certain nombre de ces clés devraient signer pour valider la transaction. Cependant, le BIP10 a été retiré et n'a jamais été intégré dans le protocole. Les fonctionnalités qu'il envisageait ont été traitées par d'autres BIP ultérieurs, tels que le BIP11 et le BIP16, qui ont introduit des mécanismes plus sophistiqués et flexibles pour les multisig sur Bitcoin.

BIP11

BIP introduit par Gavin Andresen en mars 2012 qui propose une méthode standard pour créer des transactions multi-signatures sur Bitcoin. Cette proposition permet d'améliorer la sécurité des bitcoins en exigeant plusieurs signatures pour valider une transaction. BIP11 introduit un nouveau type de script, nommé « M-of-N multisig », où « M » représente le nombre minimum de signatures requises parmi « N » clés publiques potentielles. Ce standard exploite l'opcode `OP_CHECKMULTISIG`, déjà existant dans Bitcoin, mais qui n'était pas conforme aux règles de standardisation des nœuds auparavant. Bien que ce type de script ne soit plus couramment utilisé pour des portefeuilles multisig réels, au profit du P2SH ou du P2WSH, son utilisation reste possible. Il est notamment utilisé dans des méta-protocoles tels que STAMPS. Les nœuds peuvent toutefois choisir de ne pas relayer ces transactions P2MS avec le paramètre `permitbaremultisig=0`.

De nos jours, on connait ce standard sous le nom de « bare-multisig » ou « P2MS ».

BIP12

Proposition de Gavin Andresen pour améliorer la flexibilité et la confidentialité des scripts de transaction Bitcoin. Ce BIP propose d'implémenter un nouvel opcode de script, nommé `OP_EVAL`, qui permet d'évaluer un script contenu dans les données d'un `scriptSig` lors du processus de validation de la transaction. La principale modificiation du BIP12 est de permettre l'inclusion d'un script à l'intérieur d'un autre script. Cette technique permet la création de conditions plus complexes pouvant être vérifiées lors de la dépense, sans avoir besoin de les révéler aux utilisateurs qui envoient des bitcoins vers l'adresse utilisée. Le BIP12 a été ultérieurement remplacé par d'autres propositions plus sûres, notamment le BIP16 (P2SH), qui offre une méthode différente pour atteindre les mêmes objectifs que `OP_EVAL`.

BIP13

Présente une méthode standardisée pour créer les adresses P2SH. Le BIP13 spécifie le format d'adresse P2SH, qui commence par le préfixe `3`, et qui inclut le hachage d'un script plutôt que le hachage d'une clé publique. Ce type d'adresse restera longtemps le standard préféré pour les les portefeuilles multisig.

BIP14

BIP proposé par Patrick Strateman et Amir Taaki en 2011 qui vise à distinguer les numéros de version des clients et du protocole. Le BIP14 précise la façon dont les implémentations du protocole Bitcoin doivent se présenter sur le réseau. Il suggère l'utilisation d'un format d'agent-utilisateur pour identifier la version et le type du client Bitcoin utilisé. L'objectif principal du BIP14 est de faciliter la gestion des modifications et la détection des incompatibilités entre les différents clients existants. Alors qu'il était auparavant logique de considérer le client de Satoshi comme de fait le protocole Bitcoin, la multiplication des logiciels à cette période amène le BIP14 à bien différencier les clients du protocole lui-même.

BIP16

Le BIP16 a introduit le concept de *Pay-to-Script-Hash* `P2SH` (en français « payer au hachage du script »). Proposé initialement en 2012 puis activé en 2013, le BIP16 visait à simplifier l'utilisation de transactions nécessitant des scripts complexes, tels que les transactions multisignatures, en permettant aux utilisateurs de payer vers un hash du script requis pour dépenser ces bitcoins plutôt que le script lui-même. Cette innovation a permis de réduire la quantité de données nécessaires dans la transaction initiale, déplaçant la charge de fournir le script complet vers la partie qui dépense les bitcoins. Il a également permis de ne révéler le script qu'à la dépense des bitcoins engagés sur le script, plutôt que dès la réception. Le BIP16 revêt une importance historique puisqu'il symbolise l'une des premières modifications majeures du protocole Bitcoin après le retrait de Nakamoto en 2011. Ce BIP a été le centre de débats très tendus qui ont même poussé Gavin Andresen, successeur de Satoshi en tant que mainteneur principal, à s'octroyer une période de retrait. De nombreuses autres propositions existaient et certaines ont même faillit être activées à la place du BIP16.

BIP17

Proposition de Luke Dashjr concurrente au BIP12 et au BIP16. Le BIP17 introduisait un nouvel opcode, `OP_CHECKHASHVERIFY`, conçu pour permettre la vérification d'un script présent dans le `scriptSig` face à son hachage présent dans le `scriptPubKey` avant de débloquer les fonds. Le BIP16 (P2SH) a finalement été préféré au BIP17 (CHV) suite à une période de débats intenses.

BIP21

Proposition rédigée par Nils Schneider et Matt Corallo, sur la base du BIP20 rédigé par Luke Dashjr, qui venait lui-même d'un autre document rédigé par Nils Schneider. Le BIP21 définit comment les adresses de réception doivent être encodées dans les URI (*Uniform Resource Identifier*) pour faciliter les paiements. Par exemple, une URI Bitcoin suivant le BIP21 dans laquelle je demanderait sous le label « *Pandul* » que l'on m'envoie 0.1 BTC ressemblerait à cela : `bitcoin:bc1qmp7emyf7un49eaz0nrxk9mdfrtn67v5y866fvs?amount=0.1&label=Pandul`. Cette standardisation améliore l'expérience utilisateur des transactions Bitcoin en permettant de cliquer sur un lien ou de scanner un QR code pour initier les paramètres de celles-ci. La norme BIP21 est aujourd'hui largement adoptée dans les logiciels de portefeuilles Bitcoin.

BIP22

BIP proposé en 2012 par Luke Dashjr qui introduit une méthode standardisée JSON-RPC pour les interfaces de minage externes, appelée « getblocktemplate ». Avec l'augmentation de la difficulté de minage, l'utilisation de logiciels externes spécialisés dans la production de preuves de travail s'est développée. Ce BIP propose une norme commune de communication pour le block template entre les nœuds complets et les logiciels spécialisés dans le minage. Cette méthode implique d'envoyer la structure entière du bloc, plutôt que simplement l'entête, afin de laisser la possibilité au mineur de la personnaliser.

BIP23

Ce BIP est une extension du BIP22, visant à encourager son adoption par les logiciels utilisés par les pools de minage, en particulier par le protocole Getwork, ancêtre de Stratum. Proposée par Luke Dashjr, cette extension vise à intégrer par défaut la norme du BIP22 dans Getwork, afin de faciliter son adoption par les pools de minage. L'objectif principal du BIP23 est de transmettre l'intégralité du block template aux mineurs, leur permettant ainsi d'auditer et éventuellement de modifier la structure du bloc proposé par la pool. Cette démarche vise à atténuer les préoccupations liées à la centralisation du minage, en donnant aux mineurs individuels un plus grand contrôle et une meilleure transparence sur le processus de création des blocs.

BIP30

Proposition d'amélioration impliquant un soft fork mis en œuvre le 15 mars 2012 afin de résoudre le problème des identifiants de transaction dupliqués. Avant le BIP30, il était techniquement possible d'avoir deux transactions différentes avec le même identifiant de transaction (TXID) dans la blockchain. C'est notamment arrivé deux fois pour des transactions coinbase : celle dans le bloc 91 880 dispose du même TXID que la coinbase du bloc 91 722, et celle dans le bloc 91 842 dispose du même TXID que la coinbase du bloc 91 812. Le BIP30 a résolu cette faille en imposant une nouvelle règle simple : aucune nouvelle transaction ne peut avoir le même TXID qu'une transaction précédemment enregistrée, à moins que la transaction originale n'ait été complètement dépensée (c'est-à-dire que tous ses outputs ont été utilisés). Ce soft fork a été activé grâce à la méthode du flag day. C'est donc un des premiers UASF.

BIP31

Proposition visant à améliorer les mécanismes de gestion du réseau par les nœuds Bitcoin. Avant le BIP31, les nœuds Bitcoin n'avaient pas de moyen direct de savoir si leurs pairs étaient toujours

connectés, fonctionnels et non surchargés. Le BIP31 a introduit l'utilisation d'un message `pong`, en réponse au message `ping`, qui permet une vérification active de la connectivité entre les nœuds.

BIP32

Le BIP32 a introduit le concept de portefeuille hiérarchique et déterministe (HD wallet). Cette proposition permet de générer une hiérarchie de paires de clés à partir d'une graine commune, la `master seed`, en utilisant des fonctions de dérivation à sens unique. Chaque paire de clés générée peut elle-même être la parent d'autres clés enfants, formant ainsi une structure arborescente (hiérarchique). L'avantage du BIP32 est qu'il permet de gérer de multiples paires de clés différentes avec la nécessité de ne conserver qu'une seule graine pour les régénérer. Cette innovation a notamment permis de lutter contre le phénomène de réutilisation d'adresse qui est grave pour la confidentialité des utilisateurs. Le BIP32 permet aussi la création de sous-branches au sein d'un même portefeuille afin de faciliter sa gestion.

BIP34

Soft fork appliqué en mars 2013, à partir du bloc 227 930, qui a introduit la version 2 pour les blocs Bitcoin. Cette nouvelle version exige que chaque bloc inclue dans le `scriptSig` de la transaction coinbase la hauteur du bloc en cours de création . Cette modification permet d'une part de clarifier la manière dont le réseau consent à modifier la structure des blocs et les règles de consensus. D'autre part, cela permet de forcer l'unicité de chaque bloc et de chaque transaction coinbase.

BIP35

Proposition permettant à un nœud Bitcoin d'ouvrir les informations relatives à sa mempool, c'est-à-dire les transactions en attente de confirmation. Grâce à cela, d'autres acteurs peuvent recevoir des données en temps réel sur les transactions non confirmées en adressant un message spécifique à un nœud. Avant l'adoption du BIP35, les nœuds ne pouvaient accéder qu'aux informations concernant les transactions déjà confirmées. Cette amélioration offre aux portefeuilles SPV la possibilité de recevoir des informations sur les transactions non confirmées, permet à un mineur de ne pas omettre des transactions avec des frais élevés lors d'un redémarrage, et facilite l'analyse des informations de la mempool par des services externes.

BIP37

Proposition introduite pour permettre aux portefeuilles légers (*Simplified Payment Verification*) de filtrer les transactions sans avoir à télécharger la blockchain complète. Cette méthode repose sur le concept de Bloom Filters, des structures de données probabilistes qui sont utilisées pour tester l'appartenance à un ensemble. Ces filtres permettent aux clients SPV de recevoir uniquement les transactions pertinentes pour leur portefeuille, afin de réduire la bande passante et la mémoire vive requise pour la synchronisation, notamment sur les téléphones portables. Les Bloom Filters sont transmis à un nœud complet, lequel renvoie en réponse des « Merkle blocks », contenant uniquement les transactions filtrées, l'en-tête avec la racine de Merkle, et les branches nécessaires pour associer ces transactions à la racine de l'arbre de Merkle. Le BIP37 a été critiqué pour ses lacunes en matière de confidentialité, car il expose notamment les adresses et les transactions des utilisateurs aux nœuds complets utilisés. Pour tenter de remédier à cette faille, il est possible d'intégrer des transactions supplémentaires, les « faux positifs », afin de créer du déni plausible. Néanmoins, le volume de faux positifs nécessaire pour atteindre un niveau de déni plausible satisfaisant reste considérablement élevé. Par ailleurs, le BIP37 a aussi été critiqué pour la charge de calcul imposée aux nœuds complets et pour l'introduction d'un nouveau vecteur d'attaque de type DoS. Cette option est donc par

défaut désactivée dans Bitcoin Core. Pour l'activer, il faut entrer le paramètre `peerbloomfilters=1` dans le fichier de configuration.

*Pour plus d'informations, voir la définition de **BLOOM FILTER**.*

BIP38

Proposition d'amélioration de Bitcoin qui introduit un mécanisme de chiffrement pour ajouter une protection supplémentaire sur les clés privées grâce à une passphrase. Le BIP38 permet de garantir que même si un tiers obtient physiquement la clé privée chiffrée, il ne pourra pas l'utiliser sans connaître sa passphrase. Cela ajoute une couche de sécurité supplémentaire pour protéger des bitcoins contre le vol, notamment pour la sécurité des vieux paper wallets.

Bien que l'on désigne cette proposition sous le terme de « passphrase », il est important de ne pas la confondre avec la passphrase présentée dans le BIP39, cette dernière étant d'ailleurs bien plus couramment utilisée. Le concept sous-jacent reste cependant similaire : alors que le BIP38 vise à sécuriser une clé privée individuelle, le BIP39, lui, offre une protection à l'ensemble d'un portefeuille HD.

BIP39

Le BIP39 introduit une méthode pour convertir la graine aléatoire d'un portefeuille en une suite de mots mémorisables et lisibles par l'Homme, connue sous le nom de phrase mnémonique. Cette phrase, généralement composée de 12 ou de 24 mots, permet de regénérer l'ensemble des clés privées d'un portefeuille de manière déterministe. Ainsi, au lieu de devoir mémoriser ou stocker une graine cryptographique complexe, les utilisateurs peuvent sauvegarder leurs bitcoins via une phrase de quelques mots. Le BIP39 a ainsi contribué à simplifier la gestion d'un portefeuille Bitcoin.

*Pour plus d'informations, voir la définition de **PHRASE DE RÉCUPÉRATION (MNÉMONIQUE)**.*

BIP42

Proposition d'amélioration de Bitcoin Core qui adresse un bug mineur dans le calendrier de réduction de la récompense de bloc. Cette anomalie, si elle n'était pas corrigée, aurait conduit à une création totale de bitcoins supérieure à la limite prévue des 21 millions. Plus précisément, après la fin des halvings, un nouveau cycle de création de bitcoins aurait pu théoriquement se déclencher vers l'an 2214. Le code de Core en question utilisait une opération de décalage binaire à droite sur la valeur de la récompense du mineur. Le bug provenait de l'utilisation de cette opération de décalage dans un contexte où le comportement était non défini selon les normes du langage C++. Le décalage d'un entier de 64 bits (int64_t) de plus de 63 bits à droite entre dans cette catégorie de comportement non défini. Dans le code de Bitcoin Core, cela aurait pu se produire après 64 halvings, à la hauteur de bloc n°13 440 000. Au-delà de cette limite, le comportement du décalage de bits n'était pas défini, ce qui signifie que différentes compilations pourraient interpréter le code différemment. Cela aurait pu entraîner des résultats imprévisibles, y compris la possibilité de créer une inflation infinie de bitcoins. Le BIP42 a corrigé ce problème en imposant que la récompense du bloc soit mise à zéro après 64 halvings, évitant ainsi l'utilisation d'une opération de décalage à droite dans un contexte de comportement non défini. Cette modification, qui était un soft fork, a ainsi permis de clarifier le comportement du code de Bitcoin Core. Bien que tout à fait sérieux, ce bug corrigé par le BIP42 n'était pas immédiatement critique, puisqu'il ne se serait manifesté qu'aux environs de 2214. Publié le 1er avril 2014 par Pieter Wuille, le BIP42 se distingue ainsi par son ton humoristique.

BIP43

Proposition d'amélioration qui introduit l'utilisation d'un étage de dérivation pour décrire l'objectif (« purpose field ») dans la structure des portefeuilles HD, précédement introduits dans le BIP32. Selon le BIP43, le premier niveau de dérivation d'un portefeuille HD, juste après la clé maîtresse notée `m/`, est réservé au numéro de « l'objectif » qui indique le standard de dérivation utilisé pour le reste du chemin. Ce numéro est enregistré comme un index (endurci). Par exemple, si le portefeuille suit le standard SegWit (BIP84), le début de son chemin de dérivation sera : `m/84'/`. Le BIP43 permet ainsi de clarifier les standards respectés par chaque logiciel de portefeuille pour avoir une meilleure interopérabilité entre eux. La standardisation de la suite du chemin de dérivation est décrite dans le BIP44.

BIP44

Proposition d'amélioration qui introduit une structure de dérivation hiérarchique standard pour les portefeuilles HD. LE BIP44 s'appuie sur les principes établis par le BIP32 pour la dérivation des clés et sur le BIP43 pour l'utilisation du champ « purpose ». Il introduit une structure de cinq niveaux de dérivation : `m / purpose' / coin_type' / account' / change / address_index`. Voici le détail de chaque profondeur :

- `m /` indique la clé privée maîtresse. Elle est unique pour un portefeuille et ne peut pas avoir de sœurs à la même profondeur ;
- `m / purpose' /` indique l'objectif de dérivation qui permet d'identifier le standard suivi. Ce champs est décrit dans le BIP43. Par exemple, si le portefeuille respecte le standard BIP84 (SegWit V0), l'index sera alors `84'` ;
- `m / purpose' / coin-type' /` indique le type de cryptomonnaie. Cela permet de bien différencier les branches dédiées à une cryptomonnaie, des branches dédiées à une autre cryptomonnaie sur un portefeuille multi-coin. L'index dédié au Bitcoin est le `0'` ;
- `m / purpose' / coin-type' / account' /` indique le numéro de compte. Cette profondeur permet de différencier et d'organiser facilement un portefeuille en différents comptes. Ces comptes sont numérotés à partir de `0'` . les clés étendues (`xpub` , `xprv` ...) se trouvent à ce niveau de profondeur ;
- `m / purpose' / coin-type' / account' / change /` indique la chaîne. Chaque compte tel que défini en profondeur 3 dispose de deux chaînes en profondeur 4 : une chaîne externe et une chaîne interne (également appelée « change »). La chaîne externe dérive des adresses destinées à être communiquées publiquement, c'est-à-dire les adresses que l'on nous propose lorsque l'on clique sur « recevoir » dans notre logiciel de portefeuille. La chaîne interne dérive les adresses destinées à ne pas être échangées publiquement, c'est-à-dire principalement les adresses de change. La chaîne externe est identifiée avec l'index `0` et la chaîne interne est identifiée avec l'index `1` . Vous remarquerez qu'à partir de cette profondeur, on ne réalise plus une dérivation endurcie mais une dérivation normale (il n'y a pas d'apostrophe). C'est grâce à ce mécanisme que l'on est capable de dériver l'ensemble des clés publiques enfants à partir de leur `xpub` ;
- `m / purpose' / coin-type' / account' / change / address-index` indique simplement le numéro de l'adresse de réception et de sa paire de clés, afin de la différencier de ses sœurs à la même profondeur sur la même branche. Par exemple, la première adresse dérivée dispose de l'index `0` , la deuxième adresse dispose de l'index `1` , etc...

Par exemple, si mon adresse de réception dispose du chemin de dérivation `m / 86' / 0' / 0' / 0 / 5`, on peut en déduire les informations suivantes :

- `86'` indique que nous suivons le standard de dérivation du BIP86 (Taproot ou SegWitV1) ;
- `0'` indique que c'est une adresse Bitcoin ;
- `0'` indique que l'on est sur le premier compte du portefeuille ;
- `0` indique que c'est une adresse externe ;
- `5` indique que c'est la sixième adresse externe de ce compte.

BIP47

Proposé par Justus Ranvier en 2015, ce protocole vise à résoudre le problème critique de la réutilisation des adresses Bitcoin, une pratique qui compromet gravement la confidentialité des utilisateurs sur le système. Satoshi Nakamoto, dans le White Paper de Bitcoin, avait déjà souligné l'importance d'utiliser des paires de clés distinctes pour chaque transaction afin de maintenir une ségrégation dans les différentes actions des utilisateurs. Le BIP47 introduit le concept de codes de paiement réutilisables, permettant à un utilisateur de recevoir de multiples paiements sans avoir à générer une nouvelle adresse Bitcoin manuellement pour chaque transaction. Ces codes agissent comme des identifiants virtuels, dérivés de la graine du portefeuille de l'utilisateur et situés au niveau des comptes dans la structure de dérivation d'un portefeuille HD. À partir de la combinaison des codes de paiements des 2 parties, chacune peut dériver un grand nombre d'adresses uniques appartenant à l'autre partie, sans avoir besoin de communiquer de nouveau avec elle. Le cœur de ce protocole repose sur l'algorithme ECDH (*Elliptic-Curve Diffie-Hellman*), une variante de l'échange de clés Diffie-Hellman établi sur les courbes elliptiques, qui permet aux deux parties d'établir un secret partagé pour la génération d'adresses de réception uniques. Bien que le BIP47 n'ait pas été ajouté à Bitcoin Core et ait reçu un accueil mitigé de la part de la communauté, des implémentations telles que PayNym sur Samourai Wallet et Sparrow Wallet l'ont adopté et l'ont pleinement intégré à leur écosystème d'outils de confidentialité.

BIP49

BIP informatif qui introduit la méthode de dérivation utilisée pour générer des adresses Nested SegWit dans un portefeuille HD. Le chemin de dérivation proposé suit les standards du BIP43 et du BIP44, avec l'index `49'` (dérivation renforcée) à la profondeur de l'objectif. Par exemple, la première adresse d'un compte P2SH-P2WPKH serait issue du chemin : `m/49'/0'/0'/0/0`. Les scripts Nested SegWit ont été inventé au lancement de SegWit pour faciliter son adoption. Ils permettent d'utiliser ce nouveau standard, même sur des wallets pas encore compatibles nativement avec SegWit.

*Pour plus d'informations, voir la définition de **P2SH-P2WPKH**.*

BIP50

BIP informatif faisant état d'un bug lié au passage de Berkeley DB à Level DB provoquant une division de la blockchain Bitcoin puis un réorganisation majeure de 24 blocs le 12 mars 2013. Ce BIP détaille l'incident et les actions correctives implémentées.

*Pour plus d'informations, voir la définition de **LEVELDB** et **BERKELEYDB**.*

BIP61

Introduit un message de rejet dans le protocole de communication entre les nœuds. L'objectif du BIP61 est d'ajouter un mécanisme de retour d'information lorsqu'un nœud reçoit, de la part d'un autre nœud, une transaction ou un bloc qu'il considère comme invalide. Ce message de rejet permettrait à

un nœud de signaler à l'expéditeur la raison pour laquelle cela a été rejeté. Ce type de communication devait contribuer à améliorer l'interopérabilité entre les différents clients et les communications entre les nœuds complets et les clients SPV. Les fonctionnalité amenées par le BIP61 ont finalement été abandonnées à partir de la version 0.20.0 de Bitcoin Core, car elles étaient considérées comme inutiles alors qu'elles impliquaient des besoins accrus en bande passante.

BIP65

Introduit un nouvel opcode nommé `OP_CHECKLOCKTIMEVERIFY` qui permet de rendre un UTXO inutilisable jusqu'à un moment donné dans le futur. L'application de ce BIP a nécessité un soft fork, qui est intervenu le 14 décembre 2015. Il a également introduit la version 4 des blocs.

*Pour plus d'informations, voir la définition de **OP_CHECKLOCKTIMEVERIFY**.*

BIP66

Introduit une standardisation du format des signatures dans les transactions. Ce BIP a été proposé en réaction à une divergence dans la manière dont OpenSSL gérait l'encodage des signatures sur différents systèmes. Cette hétérogénéité posait un risque de scission de la blockchain. Le BIP66 a permis d'uniformiser le format des signatures pour toutes les transactions en utilisant l'encodage DER stricte (*Distinguished Encoding Rules*). Cette modification nécessitait un soft fork. Pour son activation, le BIP66 a alors utilisé le même mécanisme que le BIP34, nécessitant l'augmentation du champ `nVersion` à sa version 3, et rejetant tous les blocs de version 2 ou inférieure une fois que le seuil de 95 % des mineurs était atteint. Ce seuil a été franchi au bloc n° 363725 le 4 juillet 2015.

BIP68

Introduit la possibilité d'utiliser des blocages temporel relatifs (*relative lock-time*) grâce au champ `nSequence`. Cela permet à une transaction de spécifier un délai relatif avant qu'elle soit incluse dans un bloc. Ce délai peut être défini en terme de nombre de bloc, ou bien comme un multiple de 512 secondes (c'est-à-dire, du temps réel). Notons que cette nouvelle interprétation du champs `nSequence` est uniquement valide si le champs `nVersion` est supérieur ou égal à `2`. Cette interprétation du champs `nSequence` se fait au niveau des règles de consensus de Bitcoin. Le timelock relatif définit un délai à partir de l'acceptation d'une transaction antérieure alors que le timelock absolu spécifie un moment précis avant lequel la transaction ne peut être incluse dans un bloc. Le BIP68 a été introduit via un soft fork le 4 juillet 2016 en même temps que le BIP112 et le BIP113, activé pour la première fois grâce à la méthode du BIP9.

*Pour plus d'informations, voir la définition de **NSEQUENCE**.*

BIP70

Protocole de paiement interactif pour Bitcoin. Il permet l'envoi de demandes de paiement et la réception de paiements de manière sécurisée et standardisée. Dans ce protocole, le client clique sur une URI Bitcoin (BIP21) étendue avec un paramètre supplémentaire (décrit dans le BIP72). La demande de paiement est signée avec le certificat SSL du commerçant. Lors de la réception et de la validation de cette demande, les détails du paiement sont automatiquement remplis dans l'interface de la transaction du portefeuille du client. Ce protocole fournit une confirmation de paiement et améliore la sécurité et l'expérience utilisateur en clarifiant l'entité bénéficiaire du paiement. Cette méthode proposée dans le BIP70 n'a jamais été largement adoptée par les commerçants.

BIP71

Définit un type de média MIME (*Multipurpose Internet Mail Extensions*), conformément à la norme RFC 2046, pour les messages de demande de paiement en bitcoins dans le BIP70. MIME est un standard Internet qui étend le format des messages électroniques pour permettre l'envoi de divers types de données de manière structurée. Dans le BIP71, l'adoption d'un type MIME spécifique pour les messages de paiement assure que les logiciels de portefeuille, lorsqu'ils envoient des messages de protocole de paiement par e-mail ou HTTP, respectent les standards Internet pour une encapsulation appropriée des messages. Cette amélioration étant groupée avec le BIP70, elle n'a jamais été largement adoptée.

BIP72

Complète le BIP70 et le BIP71 en définissant l'extension des URI Bitcoin (BIP21) avec un paramètre supplémentaire `r`. Ce paramètre permet d'inclure un lien vers une demande de paiement sécurisée et signée par le certificat SSL du commerçant. Lorsqu'un client clique sur cette URI étendue, son portefeuille contacte le serveur du commerçant pour demander les détails de paiement. Ces détails sont automatiquement remplis dans l'interface de transaction du portefeuille, et le client peut être informé qu'il paie le propriétaire du domaine correspondant au certificat de signature (par exemple, « pandul.fr »). Cette amélioration étant groupée avec le BIP70, elle n'a jamais été largement adoptée.

BIP75

Extension qui améliore le protocole de paiement BIP70 en introduisant deux innovations majeures. Premièrement, il permet à l'expéditeur d'une demande de paiement de signer volontairement cette requête et de fournir un certificat qui permet au destinataire de connaître l'identité de son interlocuteur. Secondement, il chiffre la demande de paiement retournée pour prévenir toute interception et visualisation par des tiers. Cela permet de renforcer la sécurité et la confidentialité des transactions en assurant que les détails du paiement ne sont visibles que par les participants. Le BIP75 offre aussi de nouvelle fonctionnalités pour améliorer l'expérience utilisateur. Cette amélioration étant basée sur le BIP70, elle n'a jamais été largement adoptée.

BIP78

Introduit un protocole pour utiliser des Payjoin sur Bitcoin, une structure de transaction qui améliore la confidentialité des paiements en faisant intervenir le destinataire auprès du payeur dans les inputs. Ce BIP s'inspire du BIP79, qui avait déjà présenté un concept semblable. Toutefois, ces deux BIP ne marquent pas l'origine du concept de Payjoin. En effet, l'implémentation Stowaway de Samourai Wallet existait déjà auparavant, et cette structure de transaction avait été mentionnée pour la première fois par LaurentMT en 2015, sous le nom de « *steganographic transaction* ».

*Pour plus d'informations, voir la définition de **PAYJOIN**.*

BIP84

Définit le standard de dérivation des adresses SegWit V0 (`bc1q...`) au sein d'un portefeuille déterministe et hiérarchique. Il définit l'index `84'` qui doit désormais être utilisé à la profondeur `purpose` du portefeuille HD pour les modèles de script P2WPHK.

*Pour plus d'informations, voir la définition de **BIP32**, **BIP43** et **P2WPKH**.*

BIP85

Solution pour unifier la dérivation de différents portefeuilles Bitcoin en utilisant une graine maîtresse unique pour tous. Cette proposition permet de dériver de l'entropie à partir d'une information racine pour générer plusieurs phrases mnémoniques pour plusieurs portefeuilles, sans compromettre la sécurité. L'objectif du BIP85 est de faciliter la gestion et la sauvegarde de plusieurs portefeuilles Bitcoin. Au lieu de devoir sécuriser plusieurs phrases, une seule information suffit pour toutes les autres.

BIP86

Définit le standard de dérivation des adresses SegWit V1 ou Taproot (`bc1p...`) au sein d'un portefeuille déterministe et hiérarchique. Il définit l'index `86'` qui doit désormais être utilisé à la profondeur `purpose` du portefeuille HD pour les modèles de script P2TR.

*Pour plus d'informations, voir la définition de **BIP32**, **BIP43** et **P2TR**.*

BIP90

Proposition pour simplifier le traitement de l'activation des soft forks antérieurs en remplaçant le mécanisme de signalisation par les mineurs via les numéros de version des blocs par de simples vérifications de la hauteur de bloc. Cette modification éliminerait la nécessité de vérifier les 1000 blocs précédents pour l'activation des règles de consensus, ce qui permettrait de réduire la dette technique associée à l'implémentation de ces soft fork.

BIP91

Proposition de James Hilliard (ingénieur chez Bitmain) pour faciliter l'activation du soft fork SegWit, défini dans les BIP141, BIP143 et BIP147, via un MASF sans atteindre directement le seuil requis de 95 % de la puissance de calcul signalant le soutien via le bit 1. BIP91 permet aux mineurs de signaler indirectement leur soutien à SegWit en utilisant le bit 4 dans les blocs minés. Une fois que 269 blocs sur une fenêtre de 336 blocs ont inclus le bit 4 (soit 80% de la puissance de calcul), le BIP91 se verrouille, obligeant ensuite tous les nœuds compatibles à rejeter les blocs n'incluant pas le bit 1. Cette méthode visait à rendre le BIP148 (UASF) obsolète et à éviter une scission potentielle de la blockchain le 1er août 2017. Le BIP91 a finalement été activé le 23 juillet 2017 (au bloc 477 120), juste avant la date fatidique du 1er août imposée dans le BIP148. Cela a permis de forcer le signalement de SegWit par les mineurs, qui sera finalement verrouillé le 9 août au bloc 479 808, puis activé le 24 août au bloc 481 824. Pour résumer, le BIP148 (UASF) a été créé en réaction au fait que les mineurs ne signalaient pas suffisamment SegWit, mais n'a finalement jamais été mis en œuvre. Le BIP91 (MASF) a été créé en réaction au BIP148 afin de forcer la main aux mineurs, sans pour autant risquer l'UASF du BIP148. Le BIP91 représente lui-même un soft fork, qui forcera finalement les mineurs à verrouiller le soft fork SegWit via la méthode de base (MASF BIP9).

*Pour plus d'informations, voir la définition de **MASF** et **BIP148**.*

BIP111

Propose l'ajout d'un bit de service nommé `NODE_BLOOM` pour permettre aux nœuds de signaler explicitement leur prise en charge des Bloom Filters tels que décris dans le BIP37. L'introduction de `NODE_BLOOM` permet aux opérateurs de nœuds de désactiver ce service afin de réduire les risques de DoS. L'option du BIP37 est par défaut désactivée dans Bitcoin Core. Pour l'activer, il faut entrer le paramètre `peerbloomfilters=1` dans le fichier de configuration.

*Pour plus d'informations, voir la définition de **DOS**.*

BIP112

Introduit l'opcode `OP_CHECKSEQUENCEVERIFY` (CSV) dans le langage Script de Bitcoin. Cette opération permet de créer des transactions dont la validité ne devient effective qu'après un certain délai relatif à une transaction antérieure, défini soit en nombre de blocs, soit en durée de temps. `OP_CHECKSEQUENCEVERIFY` compare la valeur en haut de la pile avec la valeur du champ `nSequence` de l'input. Si elle est supérieure et que toutes les autres conditions sont respectée, le script est valide. Ainsi, `OP_CHECKSEQUENCEVERIFY` restreint les valeurs possibles pour le champs `nSequence` de l'input qui le dépense, et ce champs `nSequence` restreint lui-même le moment où la transaction qui comprend cet input peut être incluse dans un bloc. Le BIP112 a été introduit via un soft fork le 4 juillet 2016 en même temps que le BIP68 et le BIP113, activé pour la première fois grâce à la méthode du BIP9.

*Pour plus d'informations, voir la définition de **OP_CHECKSEQUENCEVERIFY**.*

BIP113

A introduit une modification dans le calcul de toutes les opérations de timelock (`nLockTime`, `OP_CHECKLOCKTIMEVERIFY`, `nSequence` et `OP_CHECKSEQUENCEVERIFY`). Il spécifie que pour évaluer la validité des timelocks, il faut désormais les comparer au MTP (*Median Time Past*), c'est-à-dire la médiane des horodatages des 11 derniers blocs. Auparavant, on utilisait seulement l'horodatage du bloc précédent. Cette méthode rend le système plus prévisible et évite la manipulation du référentiel de temps par les mineurs. Le BIP113 a été introduit via un soft fork le 4 juillet 2016 en même temps que le BIP68 et le BIP112, activé pour la première fois grâce à la méthode du BIP9.

BIP118

Proposition d'amélioration de Bitcoin visant à introduire deux nouveaux SigHash Flag modificateurs : `SIGHASH_ANYPREVOUT` et `SIGHASH_ANYPREVOUTANYSCRIPT`. Ces fonctionnalités étendent les capacités des transactions Bitcoin, en particulier en ce qui concerne les contrats intelligents et les solutions de surcouches comme le Lightning Network. Le BIP118 permettrait notamment l'utilisation d'Eltoo. Le principal avantage du `SIGHASH_ANYPREVOUT` est de permettre la réutilisation des signatures dans plusieurs transactions, ce qui offre plus de flexibilité. Concrètement, ces SigHash permettent d'obtenir une signature qui ne couvre aucun input de la transaction.

*Pour plus d'informations, voir la définition de **SIGHASH FLAG**.*

BIP119

Introduit un nouvel opcode nommé `OP_CHECKTEMPLATEVERIFY` (CTV). CTV permettrait de créer des covenants non récursifs dans les transactions, afin d'imposer des conditions spécifiques sur la manière dont une pièce donnée peut être dépensée, y compris dans des transactions futures. Plus concrètement, il permettrait de définir des conditions sur le `scriptPubKey` des sorties d'une transaction à partir du `scriptPubKey` de l'UTXO dépensé en entrée. CheckTemplateVerify est conçu pour être simple et sans état dynamique. Sa mise en œuvre vise à étendre les capacités de script de Bitcoin pour faciliter diverses applications telles que le contrôle de congestion des transactions, la création de canaux de paiement non interactifs, les DLC, les pools de paiement… Ce nouvel opcode serait introduit en remplacement de l'`OP_NOP4`. Cette modification impliquerait un soft fork.

*Pour plus d'informations, voir la définition de **COVENANT**.*

BIP123

Établit un nouveau processus standardisé pour la classification des propositions d'amélioration de Bitcoin. Les BIP doivent dorénavant être classifiés selon 4 catégories :

- Consensus : concerne les propositions qui nécessitent un changement de consensus et affectent la compatibilité entre les versions antérieures et futures du protocole Bitcoin. Ce sont les soft forks et les hard forks ;
- Peer Services : concerne les modifications des services et des protocoles de communication entre les nœuds du réseau, sans affecter le consensus ;
- API/RPC : englobe les propositions visant à modifier les API et les RPC utilisés pour interagir avec les nœuds Bitcoin ;
- Applications : comprend les propositions d'améliorations pour les applications qui s'exécutent au-dessus du réseau Bitcoin, comme typiquement les standards liés aux logiciels de portefeuilles.

BIP125

Définit le concept de *Replace-by-Fee* (RBF), permettant à l'émetteur de remplacer une transaction non confirmée par une autre version qui inclut des frais de transaction plus élevés. Le BIP125 offre un cadre pour le signalement de RBF dans une transaction et pour son acceptation par les nœuds du réseau.

*Pour plus d'informations, voir la définition de **RBF (REPLACE-BY-FEE)**.*

BIP137

Propose un format standardisé pour signer des messages avec des clés privées Bitcoin et leurs adresses associées, afin de prouver la possession d'une adresse. Ce BIP vise à résoudre l'ambiguïté liée aux différents types d'adresses Bitcoin (P2PKH, P2SH, P2WPKH…) lors de la signature d'un message. Il introduit une méthode permettant de distinguer clairement ces formats d'adresses à travers les signatures. Ces signatures sont utiles pour diverses applications comme la preuve de fonds, l'audit, et d'autres utilisations nécessitant une authentification d'une adresse via sa clé privée. Le BIP322 a depuis introduit un nouveau format de signature qui permet d'étendre ce standard et de le généraliser pour n'importe quel type de script.

BIP141

Introduit le concept de témoin séparé (*Segregated Witness*) qui donnera son nom au soft fork SegWit. Le BIP141 introduit dans le protocole Bitcoin une modification majeure visant à résoudre le problème de malléabilité des transactions. SegWit sépare les témoins (données de signatures) du reste des données de transaction. Cette séparation est réalisée en insérant les témoins dans une structure de données distincte, engagée dans un nouvel arbre de Merkle, qui est lui-même référencé dans le bloc via la transaction coinbase, ce qui rend SegWit compatible avec les anciennes versions du protocole.

*Pour plus d'informations, voir la définition de **SEGWIT**.*

BIP143

Introduit une nouvelle manière de hacher la transaction pour la vérification des signatures dans les scripts post-SegWit. L'objectif est de minimiser les opérations redondantes lors de la vérification et

d'inclure la valeur des UTXO en entrée dans la signature. Cela résout deux problèmes majeurs de l'algorithme de hachage de transaction original :

- La croissance quadratique du hachage des données avec le nombre de signatures ;
- L'absence d'inclusion de la valeur de l'input dans la signature, ce qui posait un risque pour les hardware wallet, notamment sur le fait de connaitre les frais engagés dans la transaction.

Puisque la mise à jour SegWit, expliquée dans le BIP141, introduit une nouvelle forme de transactions dont le script ne sera pas vérifié par les vieux nœuds, le BIP143 en profite pour résoudre ce problème sans nécessiter de hard fork. Le BIP143 fait donc partie du soft fork SegWit.

BIP144

Définit de nouveaux formats de messages réseaux et de sérialisations pour la propagation des transactions et des blocs intégrant des structures de témoin séparés (SegWit). Le BIP144 établit notamment des mécanismes permettant aux pairs de signaler leur support pour SegWit et de relayer les structures de témoins sans compromettre la compatibilité avec les nœuds pas à jour.

BIP145

Met à jour l'appel JSON-RPC `getblocktemplate` pour intégrer le support de SegWit, conformément au BIP141. Cette mise à jour permet aux mineurs de construire des blocs en tenant compte de la nouvelle mesure de « poids » des transactions et des blocs introduite par SegWit, et d'autres paramètres comme le calcul de la limite des sigops.

BIP147

Proposition incluse dans le soft fork SegWit visant à résoudre un vecteur de malléabilité lié à l'élément fictif (« *dummy element* ») consommé par les opérations `OP_CHECKMULTISIG` et `OP_CHECKMULTISIGVERIFY`. En raison d'un bug off-by-one historique (erreur de décalage unitaire), ces 2 opcodes suppriment un élément supplémentaire sur la pile en plus de leur fonction de base. Pour éviter une erreur, il est donc obligatoire d'inclure une valeur factice au début du `ScriptSig` afin de satisfaire la suppression et outrepasser le bug. Cette valeur est inutile, mais elle doit forcément être là pour que le script soit valide. Le BIP11, qui a introduit le standard P2MS, conseillait de mettre un `OP_0` comme valeur inutile. Mais ce standard n'était pas imposé au niveau des règles de consensus, ce qui veut dire que n'importe quelle valeur pouvait y être placée, sans invalider la transaction. C'est en ça que `OP_CHECKMULTISIG` et `OP_CHECKMULTISIGVERIFY` contenaient un vecteur de malléabilité. Le BIP147 introduit une nouvelle règle de consensus, désignée sous le nom de `NULLDUMMY`, exigeant que cet élément fictif soit un tableau d'octets vide (`OP_0`). Toute autre valeur entraîne l'échec immédiat de l'évaluation du script. Cette modification s'applique aux scripts pré-SegWit ainsi qu'aux scripts P2WSH et nécessitait un soft fork.

*Pour plus d'informations, voir la définition de **OP_CHECKMULTISIG**.*

BIP148

Proposition introduite en mars 2017 par un développeur sous le pseudonyme de Shaolin Fry. L'objectif du BIP148 était de forcer l'activation de la mise à jour SegWit sur le protocole Bitcoin, face à la stagnation de la signalisation de ce soft fork par les mineurs via la méthode du BIP9. Le BIP148 suggérait la mise en œuvre d'un UASF (*User-Activated Soft Fork*) pour activer SegWit de force par les nœuds le 15 novembre 2017, si les mineurs n'avaient pas verrouillé SegWit d'ici le 1er août 2017. Si l'adoption de l'UASF du BIP148 avait eu lieu, les nœuds du réseau Bitcoin

auraient refusé les blocs ne signalant pas le support à SegWit, exerçant ainsi une pression sur les mineurs pour qu'ils adoptent la mise à jour. Bien que ce BIP historique n'ait finalement pas été activé, il a joué un rôle déterminant dans la réussite de l'adoption de SegWit, en contraignant les mineurs à verrouiller le soft fork via le BIP91. À plus long terme, le BIP148 a établi un précédent important, démontrant l'influence que peuvent exercer les utilisateurs via leurs nœuds complets sur les décisions de gouvernance du protocole Bitcoin.

*Pour plus d'informations, voir la définition de **UASF (USER-ACTIVATED SOFT FORK)**.*

BIP149

Proposition de Shaolin Fry pour un nouveau déploiement de SegWit (BIP141, BIP143 et BIP147) en utilisant la méthode d'activation du BIP8 avec `LOT=true`, si le déploiement initial de SegWit via le BIP9 échouait à s'activer avant le 15 novembre 2017. Contrairement à la méthode du BIP9, où un échec de signalisation entraîne l'abandon de l'activation, le BIP149 visait à activer SegWit le 4 juillet 2018, que les mineurs aient atteint le seuil de signalisation de 95% ou non. Pendant la période de huit mois entre novembre et juillet, les nœuds auraient eu la possibilité d'implémenter le BIP149, afin d'assurer une activation de SegWit par la majorité économique du réseau si l'activation par les mineurs ne se produisait pas (UASF). Une fois le premier ajustement de difficulté atteint après le 4 juillet 2018, l'activation serait passée en `LOCKED_IN`, et SegWit aurait été activé au cycle d'ajustement suivant. Contrairement au BIP148, qui prévoyait une activation de SegWit imposée par les utilisateurs ou une majorité de mineurs, le BIP149 suggérait une méthode d'activation plus progressive et mesurée, bien qu'elle demeurât résolument offensive, selon les principes du BIP8. Alors que le BIP148 laissait présager un conflit avec une séparation de la blockchain, le BIP149 écartait cette éventualité, en acceptant les blocs ne signalant pas SegWit, sauf action délibérée d'un mineur (sans incitation). Le BIP149 était donc un mécanisme d'activation de SegWit moins conflictuel que le BIP148, favorisant une adoption plus progressive et moins risquée pour le système. Ni le BIP148 ni le BIP149 n'ont finalement été mis en œuvre, SegWit ayant été activé grâce à un MASF, notamment sous l'impulsion du BIP91.

*Pour plus d'informations, voir la définition de **BIP8**, **BIP9**, **BIP91**, **BIP148**, **UASF**, **MASF** et **MÉTHODE D'ACTIVATION**.*

BIP150

Propose un mécanisme d'authentification entre les pairs sur le réseau Bitcoin pour renforcer la sécurité et garantir la propriété des nœuds. Il permet aux opérateurs de nœuds de restreindre l'accès à certains services ou d'accorder des priorités de flux de données uniquement à des pairs spécifiques, en s'authentifiant mutuellement pour éviter les attaques de type MITM. Ce BIP restera à l'état de brouillon, mais il servira d'enseignement pour le BIP324 (P2P transport V2) qui est aujourd'hui implémenté en option dans Bitcoin Core.

*Pour plus d'informations, voir la définition de **P2P TRANSPORT V2**.*

BIP151

Propose un protocole pour chiffrer les communications P2P entre pairs sur le réseau Bitcoin, afin de renforcer la sécurité et la confidentialité. Son objectif est notamment de prévenir les manipulations du trafic et les attaques de surveillance de masse. Finalement, le BIP151 a été remplacé par le BIP324 (P2P transport V2) qui est aujourd'hui implémenté en option dans Bitcoin Core.

*Pour plus d'informations, voir la définition de **P2P TRANSPORT V2**.*

BIP152

Proposition « *Compact Block Relay* » visant à réduire la bande passante nécessaire pour la transmission des blocs sur le réseau Bitcoin. Adopté en novembre 2016 dans la version 0.13.0 de Bitcoin Core, ce protocole permet de communiquer les informations des blocs de manière compacte, en se basant sur l'hypothèse que les nœuds ont déjà une grande partie des transactions d'un bloc récent dans leur mempool. Plutôt que de transmettre chaque transaction intégralement, ce qui constituerait un doublon, le BIP152 propose d'envoyer uniquement de courts identifiants pour les transactions déjà connues des pairs, accompagnés de quelques transactions sélectionnées (notamment la transaction coinbase et celles que le nœud est susceptible de ne pas connaître). Le nœud peut ensuite demander à ses pairs les éventuelles transactions manquantes. Compact Block Relay permet ainsi de diminuer la quantité de données échangées lors de la propagation des blocs, ce qui réduit ainsi les pics de bande passante et améliore l'efficacité globale du réseau.

BIP155

Proposition d'amélioration de Bitcoin introduisant un nouveau format pour les messages qui transmettent des adresses de nœuds qui acceptent des connexions entrantes. Cette proposition permet de supporter des adresses de plus grande taille, notamment pour faciliter l'intégration de protocoles réseau futurs et d'adresses IP plus complexes comme TorV3 ou I2P. Cette amélioration est également connue sous le nom de `addrv2`.

BIP156

Proposition, connue sous le nom de Dandelion, qui vise à améliorer la confidentialité du routage des transactions dans le réseau Bitcoin pour contrer la désanonymisation. Dans le fonctionnement classique de Bitcoin, les transactions sont immédiatement diffusées à de multiples nœuds. Si un observateur est en capacité de voir chaque transaction relayée par chaque nœud sur le réseau, il peut supposer que le premier nœud à envoyer une transaction est également le nœud d'origine de cette transaction, et donc que celle-ci provient de l'opérateur du nœud. Ce phénomène peut potentiellement permettre à des observateurs de lier des transactions, normalement anonymes, avec des adresses IP. L'objectif du BIP156 est de traiter ce problème. Pour ce faire, il introduit une phase supplémentaire dans la diffusion permettant de préserver l'anonymat avant la propagation publique. Ainsi, Dandelion utilise d'abord une phase de « tige » où la transaction est envoyée à travers un chemin aléatoire de nœuds, avant d'être diffusée à l'ensemble du réseau dans la phase de « capitule ». La tige et le capitule sont des références au comportement de la propagation de la transaction à travers le réseau, qui ressemble à la forme d'un pissenlit (« *a dandelion* » en anglais). Cette méthode de routage brouille la piste menant au nœud source, rendant difficile de retracer une transaction via le réseau jusqu'à son origine. Dandelion améliore donc la confidentialité en limitant la capacité des adversaires à désanonymiser le réseau. Cette méthode est d'autant plus efficace lorsque la transaction croise durant la phase de « tige » un noeud qui chiffre ses communications réseau, comme avec Tor ou P2P Transport V2. Le BIP156 n'a pour le moment pas été ajouté à Bitcoin Core.

*Pour plus d'informations, voir la définition de **P2P TRANSPORT V2**.*

BIP173

Introduit le format d'adresse Bech32 pour les adresses SegWit V0. Ce format d'adresse est caractérisé par le préfixe `bc1q`. Le format Bech32 offre plusieurs avantages :

- Il demande moins d'espace dans les codes QR ;
- Il est plus facilement interprétable par les humains ;

- Il dispose d'un mécanisme innovant pour la somme de contrôle qui est plus performant et permet de détecter et potentiellement de modifier automatiquement les fautes de frappe.

Ces caractéristiques facilitent l'utilisation des adresses de réception tout en minimisant les risques d'erreurs.

*Pour plus d'informations, voir la définition de **BECH32 ET BECH32M**.*

BIP322

Propose un nouveau standard en remplacement du BIP137 pour la signature de messages avec des clés privées Bitcoin et leurs adresses associées, afin de prouver la possession d'une adresse. Ces signatures sont utiles pour diverses applications comme la preuve de fonds, l'audit, et d'autres utilisations nécessitant une authentification d'une adresse via sa clé privée. Par rapport au BIP137, le BIP322 étend le standard de signature de messages au-delà des adresses classiques, en utilisant une approche basée sur les scripts. Il permet aux logiciels de portefeuille de signer un message pour n'importe quel script qu'ils pourraient débloquer pour dépenser des bitcoins. Pour ce faire, la méthode consiste à signer un texte en produisant une signature pour une transaction Bitcoin virtuelle. Pour les adresses P2PKH traditionnelles, le BIP322 reste compatible avec le format de signature traditionnel.

BIP324

Introduit un nouvelle version du protocole de transport Bitcoin P2P intégrant le chiffrement opportuniste pour améliorer la confidentialité et la sécurité des communications entre les nœuds. Le transport P2P V2 du BIP324 a été inclus en option (désactivé par défaut) dans la version 26.0 de Bitcoin Core, déployée en décembre 2023. Il peut être activé avec l'option `v2transport=1` dans le fichier de configuration. Cette amélioration est inspirée du BIP150 et du BIP151.

*Pour plus d'informations, voir la définition de **P2P TRANSPORT V2**.*

BIP326

Proposition d'amélioration destinée aux développeurs de logiciels de portefeuille Bitcoin prenant en charge les transactions Taproot. Son but principal est de renforcer la confidentialité des protocoles de seconde couche qui pourraient utiliser des PTLC (*Point Time Locked Contracts*), comme les CoinSwap, le Lightning Network ou les DLC (*Discreet Log Contracts*). Pour ce faire, cette proposition vise à créer du déni plausible en configurant automatiquement le champ `nSequence` des transactions Taproot de la même manière que le champ `nLocktime` était configuré dans les autres types de transactions afin de décourager les attaques de fee sniping. Autrement dit, le BIP326 demande aux logiciels de portefeuille d'utiliser le champ `nSequence` plutôt que le champ `nLocktime` pour prévenir les attaques de fee sniping, afin d'offrir une confidentialité accrue pour tous les protocoles off-chain utilisant ce champ de manière similaire. Ainsi, une transaction Taproot avec une valeur spécifique dans le champ `nSequence` pourrait être soit une dépense somme toute classique d'un portefeuille, soit une transaction de règlement d'un protocole de seconde couche avec un verrouillage temporel, rendant ces deux cas indiscernables. Si cette proposition d'amélioration est appliquée massivement par les développeurs de logiciels de portefeuille Bitcoin, cela améliorerait grandement la confidentialité et la fongibilité de Bitcoin au global.

*Pour plus d'informations, voir la définition de **FEE SNIPING**.*

BIT

Le mot « bit » est la contraction des termes « binary » et « digit » en anglais. Dans le contexte des sciences informatiques et de la cryptologie, un bit est l'unité fondamentale d'information numérique et représente la plus petite quantité d'information possible. Un bit ne peut prendre que deux valeurs distinctes : `0` ou `1`. Ces valeurs sont également appelées états binaires et peuvent représenter diverses choses, telles que les réponses `oui` ou `non`, `vrai` ou `faux` et `on` ou `off`. Les bits sont la base des systèmes numériques et sont utilisés pour stocker et transmettre de l'information dans les ordinateurs et les réseaux. Le nom de « Bitcoin » provient sûrement de la concaténation du terme « Bit », pour évoquer la nature électronique du système de paiement, et du terme « Coin », pour évoquer son objectif monétaire.

> *En français, on utilise souvent directement le mot « bit ». La traduction de ce terme anglais pourrait être « chiffre binaire ».*

Dans le contexte de Bitcoin, le terme « bit » est aussi utilisé pour désigner une subdivision monétaire du bitcoin. Un bit est égal à 100 satoshis, qui représentent la plus petite unité indivisible de bitcoin. Ainsi, un bitcoin est égal à 1 000 000 de bits ou 100 000 000 de satoshis. Cependant, l'utilisation de ce terme comme subdivision monétaire est sujet à controverse. La majorité des bitcoiners emploient soit le « sats », soit le « btc », mais pas le « bit ».

BITCOIN (B MAJUSCULE)

Bitcoin est le nom du système de cash électronique pair-à-pair créé par Satoshi Nakamoto en 2009. L'utilisation du terme Bitcoin avec un « B » majuscule peut vouloir évoquer trois choses différentes :

- Le système Bitcoin ;
- Le protocole Bitcoin ;
- Le réseau Bitcoin.

Le terme de bitcoin avec un « b » minuscule est généralement réservé pour évoquer l'unité monétaire échangée sur ce système.

BITCOIN (B MINUSCULE)

Le bitcoin (écrit avec un « b » minuscule) fait référence à l'unité monétaire utilisée pour les échanges sur le système de cash électronique Bitcoin (avec un “B” majuscule). Le bitcoin est souvent abrégé en « BTC » ou « XBT » et sert de moyen d'échange, de réserve de valeur et d'unité de compte au sein du réseau. Chaque bitcoin est divisible en 100 millions d'unités plus petites, appelées « satoshis » ou « sats », en l'honneur de son créateur, Satoshi Nakamoto. Les bitcoins sont émis par le processus de la preuve de travail (minage). Le nombre total de bitcoins est limité à 21 millions, assurant une offre finie et prévisible.

BITCOIN CASH (BCH)

Système de cryptomonnaie issu d'un hard fork de Bitcoin (BTC), réalisé le 1er août 2017 au bloc 478 558. Ce fork est survenu à la suite de désaccords au sein de la communauté Bitcoin concernant les solutions à adopter pour résoudre les problèmes de passage à l'échelle du protocole. Alors que Bitcoin a implémenté SegWit (soft fork) qui comprend une légère augmentation détournée de la capacité des blocs, Bitcoin Cash a opté pour une augmentation directe de la taille des blocs (hard fork), passant de 1 Mo à 8 Mo, avec l'objectif de réduire les frais de transaction et d'améliorer les temps de confirmation.

BITCOIN-CLI

`Bitcoin-cli`, acronyme pour « *Bitcoin Command Line Interface* », est une interface de ligne de commande conçue pour interagir avec une instance de Bitcoin Core en exécution, en particulier le daemon, `bitcoind`. Il s'agit d'un programme indépendant qui offre à l'utilisateur un moyen de communiquer et d'exécuter des commandes pour contrôler et interroger l'état de l'instance de `bitcoind`. En plus des capacités de gestion du réseau, telles que la surveillance des transactions et des blocs, `bitcoin-cli` offre également des fonctionnalités de portefeuille, permettant aux utilisateurs d'effectuer des transactions Bitcoin en envoyant et recevant des fonds.

BITCOIN.CONF

Fichier de configuration utilisé pour personnaliser le fonctionnement d'un nœud Bitcoin exécutant le client Bitcoin Core. Situé dans le répertoire de données de Bitcoin Core, ce fichier texte permet aux opérateurs de nœuds de spécifier divers paramètres et options qui influencent le comportement du nœud. Parmi les paramètres que l'on peut définir dans `bitcoin.conf`, on trouve des éléments tels que la taille de la Mempool, les restrictions sur les connexions réseau, les frais de transaction minimum de relai, ainsi que d'autres options de sécurité et de performances. La personnalisation via `bitcoin.conf` est essentielle pour adapter un nœud aux besoins spécifiques de son opérateur.

BITCOIN CORE

Bitcoin Core est le logiciel open-source de référence pour le système Bitcoin, et constitue la principale implémentation du protocole Bitcoin à ce jour. Il est développé et maintenu par un large groupe de contributeurs bénévoles. Initialement nommé « Bitcoin Qt », c'est le troisième client de l'histoire de Bitcoin après Bitcoin, de Satoshi Nakamoto, et Bitcoind. Il a été développé à partir du code original de Satoshi et a introduit une interface graphique pour l'utilisateur. Par ailleurs, encore aujourd'hui, l'interface graphique de Bitcoin Core s'appelle `bitcoin-qt`. Il est fourni avec `bitcoind` depuis la version 0.5. Le logiciel Bitcoin Core sert à plusieurs fins. Tout d'abord, il agit comme un client nœud complet. Bitcoin Core inclut également un portefeuille (wallet) pour les utilisateurs qui souhaitent stocker, gérer et effectuer des transactions directement avec Bitcoin Core.

BITCOIND

Acronyme de « *Bitcoin Daemon* ». C'est un logiciel qui implémente le protocole Bitcoin et permet aux utilisateurs d'exécuter un nœud pour des appels de procédure à distance dits RPC (« *Remote Procedure Call* »). Il s'agit d'un programme en ligne de commande (sans GUI) qui sert d'interface de communication avec Bitcoin. Autrement dit, c'est un programme qui tourne en fond avec lequel l'utilisateur peut interagir (daemon). `Bitcoind` faisait partie du client original de Satoshi Nakamoto. Certains le considèrent comme le deuxième client de l'histoire de Bitcoin, après le premier de Satoshi, puisque la version 0.2.6 du logiciel permet cette exécution comme daemon sans interface graphique. Il fut par la suite regroupé avec Bitcoin QT en 2011, client renommé par la suite « Bitcoin Core », en 2014. Aujourd'hui, `bitcoind` est donc pleinement intégré au client Bitcoin Core.

BITCOIND.PID

Fichier généré par le logiciel bitcoind (Bitcoin Daemon) lors de son exécution. Ce fichier contient l'identifiant de processus (PID) de l'instance bitcoind en cours d'exécution. Il est utilisé pour suivre et gérer le processus du logiciel, permettant à d'autres applications ou scripts de l'identifier facilement et d'interagir avec lui si nécessaire.

BITCOIN FOG

Service de mixage centralisé qui a opéré durant le début de la décennie 2010. Bitcoin Fog offrait aux utilisateurs la possibilité d'accroître leur confidentialité en regroupant leurs bitcoins dans une transaction unique, dans le but de dissocier les pièces de leur historique de transactions. Étant donné sa nature centralisée, les utilisateurs devaient faire confiance à l'opérateur du service pour ne pas détourner les fonds et pour ne pas conserver de trace des opérations de mixage.

BITCOIN GOLD (BTG)

Système de cryptomonnaie créé à partir d'un hard fork de Bitcoin (BTC), qui a eu lieu le 24 octobre 2017 au bloc 491,407. Le fork a été initié par un groupe de développeurs et d'investisseurs mécontents de la direction prise par Bitcoin, en particulier en ce qui concerne la concentration croissante de la puissance de minage entre les mains de quelques grandes fermes utilisant des ASIC. Bitcoin Gold voulait démocratiser le processus de minage en utilisant un nouvel algorithme de preuve de travail, Equihash, qui est résistant aux ASIC et favorise donc le minage par des cartes graphiques (GPU). Le but était de rendre la participation au minage accessible à un plus grand nombre de personnes afin de le décentraliser et de réduire les risques liés à la centralisation. BTG utilisait initialement le même algorithme de minage que Zcash (ZEC). Ils l'ont ensuite légèrement modifié pour créer Equihash.

> *Attention, l'altcoin Bitcoin Gold (BTG) ne doit pas être confondu avec bit gold, le concept de Nick Szabo.*

BITCOIN KNOTS

Implémentation du protocole Bitcoin. Bitcoin Knots est une alternative au logiciel de référence Bitcoin Core, proposant quelques règles et fonctionnalités différentes, tout en étant compatible avec les autres nœuds. Knots est développé et maintenu par Luke Dashjr.

BITCOIN INQUISITION

Fork logiciel de Bitcoin Core qui vise à tester l'intégration de nouvelles propositions d'amélioration de Bitcoin dans un environnement contrôlé sur des réseaux de test comme un Signet. Il inclut par exemple le support pour le BIP118 (ANYPREVOUT) et le BIP119 (CHECKTEMPLATEVERIFY). Le but de Bitcoin Inquisition est d'expérimenter, afin d'avoir une meilleure compréhension des applications, des avantages, des risques et des compromis liés à ces modifications. Bitcoin Inquisition est maintenu par Anthony Towns.

BITCOIN JESUS

Surnom donné à l'entrepreneur et investisseur Roger Ver, qui a été un promoteur précoce de Bitcoin. Ce surnom lui a été donné en raison de son rôle influent dans la popularisation de Bitcoin au début des années 2010. Il est également connu pour son soutien au hard fork Bitcoin Cash (BCH), qu'il considère comme plus fidèle à la vision originale de Bitcoin.

BITCOIN POOLED MINING (BPM)

Autre nom donné à la méthode « SCORE BASED METHOD » pour le calcul de la rémunération des mineurs dans le contexte des pools de minage.

> *Pour plus d'informations, voir la définition de **SCORE (SCORE BASED METHOD)**.*

BITCOIN QT

Bitcoin QT est un client Bitcoin intégrant une interface graphique publié en mai 2011. Il s'inscrit dans la lignée du client de Satoshi lui-même. En 2014, Bitcoin QT est renommé « Bitcoin Core ». C'est aujourd'hui l'implémentation de référence du protocole Bitcoin. Il est fourni avec `bitcoind` depuis la version 0.5. Par ailleurs, encore aujourd'hui, l'interface graphique de Bitcoin Core s'appelle `bitcoin-qt` en référence aux origines du logiciel.

> *« QT » provient du nom de la bibliothèque utilisée pour l'interface graphique, qui s'appelle donc « QT ». Le nom « Qt » est parfois interprété comme un jeu de mots sur la sonorité du terme « cute » (mignon en anglais).*

BITCOIN SATOSHI VISION (BSV)

Système de cryptomonnaie issu d'un hard fork de Bitcoin Cash (BCH), lui-même dérivé de Bitcoin (BTC). Le fork de Bitcoin SV s'est produit le 15 novembre 2018 au bloc 556 766 en raison de désaccords au sein de la communauté Bitcoin Cash, notamment concernant la taille des blocs et la supposée vision de Satoshi Nakamoto, le créateur de Bitcoin. Deux camps se sont affrontés :

- Les partisans du fork « Bitcoin Cash ABC », qui est devenu Bitcoin Cash (BCH). Ce groupe était notamment soutenu par le célèbre entrepreneur Roger Ver ;
- Les promoteurs du fork « Bitcoin Cash Satoshi Vision », qui a abouti à la création de Bitcoin Satoshi Vision (BSV). Ce camp était entre autres soutenu par Craig Wright.

Bitcoin SV se distingue de Bitcoin Cash, et encore plus de Bitcoin, par sa limite de taille de bloc considérablement élevée. Cette spécificité vise à s'aligner sur ce que ses partisans considèrent être la vision initiale de Satoshi Nakamoto pour Bitcoin.

BITCOINTALK

Forum en ligne dédié aux discussions sur Bitcoin. Introduit le 22 novembre 2009 par Satoshi Nakamoto, ce forum prend la suite de l'espace de discussion dédié à Bitcoin sur sourceforge.net. BitcoinTalk sert de plateforme pour l'échange d'informations, de nouvelles, de débats techniques et d'analyses.

BIT GOLD

Système d'or numérique décentralisé conceptualisé par Nick Szabo en 1998 puis publié en 2005. Bit gold été conçu pour générer et échanger une ressource virtuelle appelée le bit gold. Ce système ne reposait sur aucun bien physique, mais visait à créer une forme de rareté infalsifiable. Le protocole bit gold reposait sur la création monétaire par preuve de travail, où les morceaux de bit gold étaient créés via la puissance de calcul des ordinateurs, formant ainsi une chaîne de preuve de travail. Chaque preuve de travail était horodatée puis ajoutée à un registre de propriété. La vérification et le transfert de la propriété de bit gold étaient effectués via un registre public, où les utilisateurs étaient identifiés par des clés publiques. Bit gold est resté à l'état de concept et n'a jamais été implémenté. Ce système est clairement un des précurseurs de Bitcoin avec b-money et RPoW, mais Satoshi semblait ne pas connaître son existence avant la création de Bitcoin. Il y fera mention plus tard sur le forum BitcoinTalk.

BITVM

Protocole introduit par Robin Linus en 2023, qui vise à étendre les capacités de développement applicatif de Bitcoin. BitVM permet de réaliser n'importe quelle opération de calcul de manière arbitraire et d'utiliser ce calcul pour diriger les bitcoins engagés. Le protocole consiste à déplacer toute les calculs en dehors de la chaîne tout en permettant de contester le calcul sur la chaîne si l'autre partie prétend à un résultat frauduleux. BitVM procure ainsi à Bitcoin une capacité de calcul quasi Turing-complet, et ce, sans requérir aucune modification au niveau du consensus. BitVM reproduit le comportement d'une porte logique `NAND` grâce à une utilisation conjointe des opcodes `OP_BOOLAND` (qui reproduit lui-même le comportement d'une porte logique `AND`) et `OP_NOT` (qui reproduit le comportement d'une porte logique `NOT`). Justement, cette porte logique `NAND` peut être utilisée à la chaîne pour reproduire le comportement de toutes les autres portes logiques existantes. C'est ce que l'on appelle une « porte universelle ». Par extension, une suite de porte logique `NAND` peut donc reproduire n'importe quel circuit de calcul. L'idée avec BitVM est de stocker ces suites de calculs `NAND` comme des feuilles dans le MAST d'une transaction Taproot.

BLK????.DAT

Nom des anciens fichiers utilisés dans Bitcoin Core pour stocker les données brutes des blocs de la blockchain. Ces fichiers ont été remplacés par les fichiers blocks/blk?????.dat depuis la version 0.8.0.

BLKINDEX.DAT

Nom de l'ancien fichier utilisé dans Bitcoin Core pour stocker diverses informations sur la blockchain, remplacé depuis la version 0.8.0 par les fichiers dans chainstate/, blocks/index/ et blocks/rev?????.dat.

BLKTREE/

Nom de l'ancien dossier utilisé dans Bitcoin Core pour cataloguer les métadonnées sur tous les blocs. Ce fichier a été remplacé par le dossier blocks/index/ dans la version 0.8.0.

BLOC

Un bloc est une structure de données dans le système Bitcoin. Un bloc contient un ensemble de transactions valides et des métadonnées contenues dans son entête. Chaque bloc est lié au suivant par le hachage de son entête, formant ainsi la blockchain (chaîne de blocs). La blockchain agit comme un serveur d'horodatage qui permet à chaque utilisateur de connaître l'ensemble des transactions passées, afin de vérifier la non-existence d'une transaction et éviter la double dépense. Les transactions sont organisées dans un arbre de Merkle. Cet accumulateur cryptographique permet de produire un condensat de toutes les transactions d'un bloc, appelé « Racine de Merkle » (Merkle root). L'entête d'un bloc contient 6 éléments :

- La version du bloc ;
- L'empreinte du bloc précédent ;
- La racine de l'arbre de Merkle des transactions ;
- L'horodatage du bloc ;
- La cible de difficulté ;

- Le nonce (« *Number only used ONCE* »).

Pour être valide, un bloc doit disposer d'un entête qui, une fois haché avec `SHA256d`, produit un condensat inférieur ou égal à la cible de difficulté.

BLOC CANDIDAT

Un bloc candidat est un bloc en cours de création par un mineur participant au processus de minage du système Bitcoin. Le bloc candidat est une structure de données temporaire qui contient des transactions en attente d'être confirmées, mais ne dispose pas encore d'une preuve de travail valide (proof-of-work) pour être ajouté à la blockchain. Le mineur sélectionne les transactions à inclure dans le bloc candidat en fonction de divers facteurs, tels que les frais de transaction associés et les contraintes de taille de bloc. Une fois les transactions sélectionnées, le mineur génère l'entête du bloc, qui comprend la version, un condensat des transactions (racine de Merkle), un horodatage, le hash du bloc précédent, la cible de difficulté et un nonce. Le mineur tente ensuite de trouver un hash de son entête satisfaisant la difficulté cible du moment. Pour ce faire, il modifie le nonce présent dans l'entête. Il peut également modifier d'autres informations présentes dans son bloc candidat. C'est le mécanisme de la preuve de travail. Si le mineur réussit à trouver un hash valide, le bloc candidat devient un bloc valide et est diffusé au réseau pour être ajouté à la blockchain.

BLOCKCHAIN

La blockchain est le nom communément donné au serveur d'horodatage distribué du système Bitcoin. C'est une chaîne de blocs. Chaque bloc est lié au suivant par son empreinte cryptographique. Pour éviter la double dépense sur Bitcoin sans recourir à une autorité centrale, il faut que chaque utilisateur vérifie la non-existence d'une transaction dans le système. Le seul moyen de s'assurer de l'absence d'une transaction est d'être au courant de toutes les transactions Bitcoin passées. Dans cet objectif, les transactions sont horodatées au sein de blocs, et chaque utilisateur dispose de l'entièreté de la blockchain.

> *Suite aux nombreuses utilisations marketing abusives du terme de « Blockchain », notamment à la fin des années 2010, beaucoup de bitcoiners refusent l'emploi de ce mot. Certains préfèrent parler de « TimeChain » pour évoquer ce concept. D'autres, se référant au White Paper de Satoshi Nakamoto, évoquent une « Proof-of-Work Chain ». En français, le terme anglais de « Blockchain » est globalement admis. On peut également utiliser la traduction « chaîne de blocs ».*

BLOCKS INDEX

Structure de données LevelDB dans Bitcoin Core qui catalogue des metadonnées sur tous les blocs. Chaque entrée dans cet index renseigne des détails tels que l'identifiant du bloc, sa hauteur dans la blockchain, le pointeur vers le bloc dans la base de données, et d'autres métadonnées. Cette indexation permet de trouver rapidement un bloc stocké en mémoire.

BLOCKS/BLK?????.DAT

Nom des fichiers dans Bitcoin Core qui stockent les données brutes des blocs de la blockchain. Chaque fichier est identifié par un numéro unique dans son nom. Ainsi, les blocs sont enregistrés dans l'ordre chronologique, en commençant avec le fichier blk00000.dat. Lorsque ce fichier atteint sa capacité maximale, les blocs suivants sont enregistrés dans blk00001.dat, puis blk00002.dat, et ainsi de suite. Chaque fichier blk a une capacité maximale de 128 mébioctets (MiB), ce qui équivaut à un peu plus de 134 mégaoctets (Mo).

BLOCKS/INDEX/

Voir la définition de **BLOCKS INDEX**.

BLOCKS/REV?????.DAT

Nom des fichiers dans Bitcoin Core qui stockent les informations nécessaires pour éventuellement annuler les modifications apportées à l'UTXO set par les blocs précédemment ajoutés. Chaque fichier est identifié par un numéro unique qui est le même que le fichier blk?????.dat auquel il correspond. Les fichier rev?????.dat contiennent les données d'annulation correspondant aux blocs stockés dans les fichiers blk?????.dat. Ces données sont essentiellement une liste de tous les UTXO dépensés en input dans un bloc. Ces fichiers d'annulation permettent au nœud de revenir à un état antérieur en cas de réorganisation de la blockchain provocant l'abandon de blocs préalablement valides.

BLOCKSIGNERS

Dans le contexte de Liquid (sidechain de Bitcoin), ce sont les entités responsables de la construction et de la validation des blocs au sein de cette chaîne. Liquid utilise un modèle de fédération où les blocksigners, sélectionnés parmi les membres de la fédération, opèrent conjointement pour confirmer les transactions et créer de nouveaux blocs afin de former le consensus de la sidechain. Le rôle de blocksigner fait partie des fonctionnaires dans Liquid. Ces derniers assurent à la fois ce rôle, et celui de watchmen (gardien).

En français, on peut traduire « blocksigners » par « signataires de blocs ».

BLOCKSTREAM

Entreprise spécialisée dans le développement de solutions autour de Bitcoin. Blockstream est à l'initiative de la sidechain Liquid, de l'implémentation du Lightning Network Core Lightning ou encore des portefeuilles Jade et Green. Elle est également connue pour employer des développeurs Bitcoin Core. La société Blockstream est actuellement dirigée par le cypherpunk et cryptographe Adam Back, l'inventeur de Hashcash, le protocole qui a inspiré la preuve de travail sur Bitcoin.

BLOCK TEMPLATE

Ensemble d'informations fournies par un nœud Bitcoin à un logiciel de minage ou une pool, contenant les données nécessaires pour construire un nouveau bloc candidat. Cette structure inclut les transactions sélectionnées pour être incluses et l'entête du bloc. Une fois que le mineur reçoit ces informations, il est en capacité de commencer à chercher une preuve de travail valide pour le bloc candidat.

BLOCK WITHHOLDING

Attaque spécifique au minage dans une pool. C'est une pratique malveillante où un participant de la pool trouve un bloc avec une preuve de travail valide mais ne le partage pas avec la pool. L'attaquant soumet des preuves de travail partielles pour maintenir l'apparence de participation active, mais retient la preuve de travail valide, privant ainsi la pool des récompenses du bloc concerné. Cette tactique vise à diminuer les gains de la pool sans en tirer de bénéfice direct, affectant ainsi la rentabilité de celle-ci.

BLOOM FILTER

Structure de données probabiliste utilisée pour tester si un élément fait partie d'un ensemble. Les Bloom Filters permettent de vérifier rapidement l'appartenance sans tester l'ensemble des données. Ils sont particulièrement utiles dans les contextes où l'espace et la vitesse sont critiques, mais où un taux d'erreur faible et contrôlé est acceptable. En effet, les Bloom Filters ne produisent pas de faux négatifs, mais ils produisent une certaine quantité de faux positifs. Lorsqu'un élément est ajouté au filtre, plusieurs fonctions de hachage génèrent des positions dans un tableau de bits. Pour vérifier l'appartenance, les mêmes fonctions de hachage sont utilisées. Si tous les bits correspondants sont définis, l'élément est probablement dans l'ensemble, mais avec un risque de faux positifs. Les filtres de Bloom sont largement utilisés dans le domaine des bases de données et des réseaux. On sait notamment que Google les utilise pour son système de gestion de base de données compressées *BigTable*. Dans le protocole Bitcoin, on les utilise notamment pour les portefeuilles SPV selon le BIP37.

> *Lorsque l'on parle spécifiquement de l'utilisation des Bloom Filters dans le cadre des transactions Bitcoin, on retrouve parfois le terme de « Transaction Bloom Filtering ». Pour plus d'informations, voir la définition de* ***BIP37****.*

B-MONEY

Prototype de cryptomonnaie décentralisée conçue par Wei Dai en 1998. Ce système imaginait un réseau où les participants seraient identifiés uniquement par des clés publiques, et où chaque transaction seraient signée par l'expéditeur. B-money était établi sur un modèle de comptes plutôt que d'UTXO, comme pour Bitcoin actuellement. Il permettait la création de monnaie par une sorte de preuve de travail liée à un panier de marchandises. C'était donc un précurseur au principe actuel de stablecoin. Ce concept n'a jamais été mis en œuvre.

BOLT

Sigle de « *Basis Of Lightning Technology* ». C'est une série de spécifications destinées à permettre l'interopérabilité de Lightning entre les différentes implémentations de ce protocole de seconde couche. Ces spécifications détaillent les règles et les normes à respecter afin que les nœuds Lightning forment un seul et même réseau.

BOUTISME

Traduction française de « endianness ».

> *Pour plus d'informations, voir la définition de* ***ENDIANNESS****.*

BRANCH-AND-BOUND

Méthode utilisée pour la sélection de pièces dans le portefeuille de Bitcoin Core depuis la version 0.17 et inventée par Murch. Le BnB est une recherche pour trouver un ensemble d'UTXO qui correspond au montant exact des sorties à satisfaire dans une transaction, afin de minimiser le change et les frais associés. Son but est de réduire un critère de gaspillage qui prend en compte à la fois le coût immédiat et les coûts futurs prévus pour le change. Cette méthode est plus précise en termes de frais comparée aux heuristiques antérieures comme le Knapsack Solver. Le Branch-and-Bound est inspiré de la méthode de résolution de problème de même nom, inventée en 1960 par Ailsa Land et Alison Harcourt.

> *Cet méthode est également parfois nommée « Algorithme de Murch ».*

BRANCHE

Dans le cadre de Git, représente une séparation du flux de travail principal, permettant le développement en parallèle, sans affecter la branche principale (généralement nommée `master` ou `main`). Les branches facilitent les modifications, les tests et les expérimentations dans un environnement isolé, avant leur éventuelle intégration dans le projet principal via un merge (fusion).

BRC-20

BRC-20 définit un ensemble de règles et de méthodes à respecter pour permettre une interaction avec des jetons non natifs sur Bitcoin. Il s'appuie sur les inscriptions du protocole Ordinals afin de définir des fonctions interprétées en dehors de la chaine. Ce standard a été créé par le développeur Domo, au début du mois de mars 2023. Selon son créateur, ce standard n'est qu'une expérimentation. Cela n'a pas empêché la machine spéculative de prendre le dessus durant les mois d'avril et de mai 2023. Des milliers d'investisseurs se sont emparés de ce standard, en achetant massivement les jetons BRC-20, créant au passage une hausse soudaine et historique des frais de transaction sur Bitcoin. Les jetons BRC-20 sont dénués d'existence concrète sur Bitcoin. Ils sont off-chain. Ce protocole utilise simplement Bitcoin, à travers le protocole Ordinals, pour stocker et horodater des fonctions permettant la gestion des jetons BRC-20. Ces fonctions sont encodées dans un format texte JSON, puis elles sont diffusées sous forme d'inscription Ordinals sur Bitcoin. Il en existe trois :

- `deploy` , qui permet de créer un nouveau jeton BRC-20 et de définir ses conditions d'utilisation ;
- `mint` , qui permet de réclamer des jetons BRC-20 spécifiques. Cela représente leur émission ;
- `transfer` , qui permet de transférer des jetons BRC-20 entre plusieurs utilisateurs.

Pour exécuter ce protocole, il faut que des personnes maintiennent des serveurs qui recensent l'intégralité des fonctions. Le standard BRC-20 est alors une utilisation très peu optimisée de Bitcoin par rapport à un protocole tel que RGB.

BTC

Symbole boursier ou monétaire (ticker) utilisé pour représenter une unité de bitcoin sur les plateformes d'échange. Il sert à identifier rapidement le bitcoin parmi d'autres actifs et monnaies. Une unité de bitcoin (1 BTC) est égale à 100 000 000 de satoshis (ou « sats »).

BTCPAY SERVER

Processeur de paiement open-source qui permet aux commerçants et aux utilisateurs d'accepter des paiements en bitcoins sans dépendre d'un tiers pour le traitement des transactions. Lancé en 2017, BTCPay Server offre une solution d'intégration de paiements en cryptomonnaies pour les sites e-commerce, avec des fonctionnalités avancées comme le support de hardware wallets, des outils de facturation et de comptabilité, ainsi que la compatibilité avec le Lightning Network. Son développement a été initié par Nicolas Dorier, en réaction aux actions de Bitpay qui, selon lui, avaient induit en erreur ses utilisateurs en les poussant vers l'adoption de SegWit2x, considéré à tort comme le "vrai" bitcoin. Cette opposition s'est cristallisée dans un tweet désormais célèbre de Nicolas Dorier en août 2017 : « *This is lies, my trust in you is broken, I will make you obsolete* ».

C

C

Langage de programmation de haut niveau, créé dans les années 1970 par Dennis Ritchie. Il est connu pour sa performance, sa flexibilité et sa portabilité, ce qui en fait un choix populaire pour le développement de logiciels. Sa syntaxe a servi de base à de nombreux autres langages, y compris C++, Java et C#.

C# (C SHARP)

Langage de programmation moderne, orienté objet, développé par Microsoft. Il est conçu pour être simple mais puissant.

C++

Langage de programmation polyvalent, évoluant du C, connu pour sa puissance et sa flexibilité. Utilisé pour le développement logiciel complexe, il prend en charge la programmation orientée objet et offre de riches fonctionnalités pour la gestion de la mémoire et des ressources système.

CAHOOTS

Dans le cadre du portefeuille Samourai Wallet et des autres logiciels de portefeuilles qui l'implémente, un Cahoot désigne tous les types de transactions réalisées en collaboration entre plusieurs utilisateurs. Procéder à un Cahoot signifie donc participer conjointement à une transaction. Cette collaboration s'articule autour de l'échange de transactions partiellement signées. Ces échanges peuvent se faire soit manuellement, via des codes QR, soit de manière automatisée, via le réseau de communication Soroban.

- Les transactions Stowaway (Payjoin) ;
- Les transactions Stonewall x2 ;
- Les transactions Joinbot.

CANAL DE PAIMENT

Dans le cadre du Lightning Network, un canal de paiement est une connexion bidirectionnelle entre deux nœuds Lightning et qui permet de faire des échanges de bitcoins off-chain. On-chain, un canal de paiement est représenté par une adresse multi-signatures 2/2 détenue par les deux participants. Le canal de paiement nécessite une transaction on-chain pour son ouverture et une transaction off-chain pour sa fermeture. Entre ces deux évènements, les utilisateurs du canal peuvent réaliser un très grand nombre d'échanges de bitcoins off-chain, sur le Lightning Network, sans nécessiter une activité on-chain. Sur Lightning, il est possible de router un paiement à travers plusieurs canaux de paiements et plusieurs nœuds, afin d'envoyer des bitcoins sans forcément ouvrir un canal direct avec le receveur.

CAPACITÉ DE CANAL LIGHTNING

Quantité de bitcoins bloqués sur une adresse multisignatures qui représente un canal de paiement sur le Lightning Network. La capacité d'un canal est donc la quantité maximale de sats qui peut être transmise via ce canal spécifique. Elle est définie au moment de la création du canal par la somme des fonds qu'une partie engage dans le canal. L'« *inbound capacity* », ou « capacité entrante », désigne la quantité maximale de bitcoins qu'un nœud peut recevoir via un canal. L'« *outbound capacity* », ou

« capacité sortante » représente la quantité maximale de bitcoins qu'un nœud peut envoyer à travers un canal spécifique.

CASHU

Protocole open-source de monnaie électronique chaumienne, similaire au système eCash de David Chaum, mais qui fonctionne sur Bitcoin et le Lightning Network. Plus précisément, Cashu est inspiré d'une variante d'eCash proposée en 1996 par David Wagner nommée « *Chaumian ecash without RSA* ». Cashu peut être utilisé sur des portefeuilles custodiaux afin que le serveur ne puisse identifier ni les propriétaires des fonds, ni les détails des transactions, offrant ainsi une amélioration de la confidentialité. Les utilisateurs peuvent générer des jetons Cashu en échange de bitcoins, qui sont signés par le serveur sans connaitre l'utilisateur. Les jetons peuvent ensuite être transférés entre utilisateurs de manière instantanée, privée et sans frais.

*Pour plus d'informations, voir la définition de **ECASH**.*

CET

Sigle de « *Contract Execution Transaction* ». C'est une transaction spécifique au sein d'un DLC qui permet le règlement final entre les parties en fonction de l'issue d'un événement futur. Lorsque l'oracle publie une signature correspondant au résultat de l'événement, les parties utilisent cette signature pour compléter et déverrouiller la CET qui envoie les fonds à la partie gagnante. La CET signée est ensuite minée, et le gagnant reçoit les bitcoins qui lui sont dus selon les conditions du contrat intelligent. Toutes les autres CET potentielles, qui auraient été exécutées en cas de résultats différents, deviennent obsolètes et sont abandonnées.

*Pour plus d'informations, voir la définition de **DLC (DISCREET LOG CONTRACT)**.*

CHANNEL FACTORIES

Mécanisme avancé en cours de travail sur Lightning, permettant la création et la gestion de plusieurs canaux de paiement à partir d'un seul UTXO. Les channel factories utilisent des adresses multisig `n-of-n` pour qu'un groupe d'utilisateurs puisse détenir collectivement un seul UTXO. De là, ils peuvent ouvrir et fermer des canaux de paiement entre eux sans transactions supplémentaires on-chain, sauf lorsqu'ils souhaitent retirer leurs fonds de la factory. Cette méthode permettrait de réduire considérablement les coûts et l'espace occupé sur Bitcoin pour des transactions Lightning. En pratique, cela signifie que des opérations qui nécessiteraient normalement des transactions on-chain pour chaque ouverture ou fermeture de canal peuvent être effectuées hors chaîne, avec la sécurité garantie par la capacité de publier les transactions non-publiées si nécessaire. Pour reprendre les mots de David A. Harding, les channel factories peuvent être décrites comme des canaux Lightning utilisés pour générer d'autres canaux Lightning.

CHAINSTATE/

Nom technique donné au dossier utilisé pour stocker l'UTXO set sur Bitcoin Core. C'est donc en réalité un synonyme d'« UTXO set ».

*Pour plus d'informations, voir la définition de **UTXO SET**.*

CHARGE UTILE (PAYLOAD)

Dans le contexte général de l'informatique, une charge utile désigne les données essentielles transportées dans un paquet de données plus large. Par exemple, dans une adresse SegWit V0 sur Bitcoin, la charge utile correspond au hachage de la clé publique, sans les diverses métadonnées (le HRP, le séparateur, la version de SegWit et la somme de contrôle). Par exemple, sur l'adresse `bc1qc2eukw7reasfcmrafevp5dhv8635yuqays50gj`, nous avons :

- `bc` : la partie lisible par l'homme (HRP) ;
- `1` : le séparateur ;
- `q` : la version de SegWit. Ici, c'est la version 0 ;
- `c2eukw7reasfcmrafevp5dhv8635yuqa` : la charge utile, ici, le hachage de la clé publique ;
- `ys50gj` : la somme de contrôle.

CHAUMIAN COINJOIN

Amélioration du concept de coinjoin, introduit pour la première fois en 2013 par Gregory Maxwell, qui utilise les signatures aveugles de Chaum pour renforcer l'anonymat des transactions coinjoins. Dans ce protocole, les utilisateurs soumettent des entrées et une adresse de réception cryptographiquement aveuglées à un coordinateur. Cette adresse privée est destinée à recevoir les bitcoins en sortie de coinjoin. Le coordinateur signe ces tokens et les renvoie aux utilisateurs. Les utilisateurs se reconnectent ensuite de manière anonyme au serveur du coordinateur et révèlent ensuite leurs adresses de sortie en clair pour la construction de la transaction. Le coordinateur peut vérifier que toutes ces adresses de réception proviennent bien d'utilisateurs légitimes, puisqu'il a signé leur version aveuglée auparavant. En revanche, il ne peut pas associer une adresse de sortie spécifique à un utilisateur donné en entrée. Il n'y a donc aucun lien entre les entrées et les sorties, même du point de vue du coordinateur. Cette méthode garantit que le coordinateur ne peut ni compromettre l'anonymat des participants, ni voler les bitcoins durant tout le processus de coinjoin.

*Pour plus d'informations, voir la définition de **COINJOIN**.*

CHIFFRER (CHIFFREMENT)

Méthode cryptographique permettant de convertir une information brute en information chiffrée. Une information chiffrée masque la signification originale des données pour empêcher qu'elles ne soient connues. Le chiffrement consiste en une série de transformations effectuées sur l'information originale à l'aide d'une clé. Si ces transformations sont réversibles, le processus d'inversion correspondant est appelé « déchiffrement », et il permet de restaurer les informations à leur état brut.

CIBLE DE DIFFICULTÉ

Le facteur de difficulté, aussi connu sous le nom de cible de difficulté, est un paramètre crucial dans le mécanisme de consensus par preuve de travail (Proof of Work, PoW) utilisé par Bitcoin. La cible représente une valeur numérique qui détermine la difficulté pour les mineurs de résoudre un problème cryptographique spécifique, appelé preuve de travail, lors de la création d'un nouveau bloc dans la blockchain. La cible de difficulté est un nombre ajustable de 256 bits (64 octets) déterminant une limite d'acceptabilité pour le hachage de l'entête des blocs. Autrement dit, pour qu'un bloc soit valide, le hachage de son entête avec `SHA256d` (double `SHA256`) doit être numériquement inférieur ou égal à la cible de difficulté. La preuve de travail consiste à modifier le champ `nonce` de l'entête du bloc ou de la transaction coinbase jusqu'à ce que le hachage résultant soit inférieur à la valeur cible. Cette

cible est ajustée tous les 2016 blocs (environ toutes les deux semaines), lors d'un évènement que l'on appelle « ajustement ». Le facteur de difficulté est recalculé en fonction du temps qu'il a fallu pour miner les 2016 blocs précédents. Si le temps total est inférieur à deux semaines, la difficulté augmente en ajustant la cible à la baisse. Si le temps total est supérieur à deux semaines, la difficulté diminue en ajustant la cible à la hausse. L'objectif est de conserver un temps de minage par bloc moyen à 10 minutes. Ce temps entre chaque bloc permet d'éviter les divisions du réseau Bitcoin, résultant en un gaspillage de la puissance de calcul. La cible de difficulté se trouve dans chaque entête de bloc, au sein du champ `nBits`. Puisque ce champ est réduit à 32 bits et que la cible fait en réalité 256 bits, la cible est compactée dans un format scientifique moins précis.

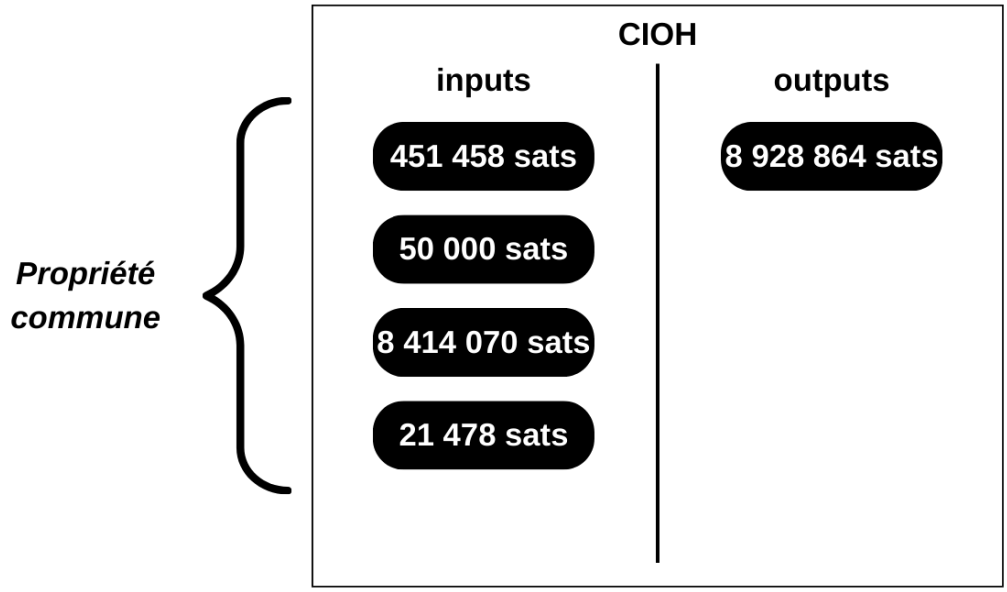
> *La cible de difficulté est parfois également nommée « facteur de difficulté ». Par extension, on peut l'évoquer avec son encodage dans les entêtes de bloc avec le terme « nBits ».*

CIOH

Sigle de « *Common Input Ownership Heuristic* ». C'est une heuristique utilisée dans le domaine de l'analyse et du traçage des transactions sur Bitcoin qui suppose que toutes les entrées d'une transaction appartiennent à une même entité ou à un même utilisateur. Lorsque l'on observe les données publiques d'une transaction Bitcoin, et que l'on y repère plusieurs entrées (inputs), alors, s'il n'y a pas de paternes où d'autres informations qui viendraient infirmer cela, on peut estimer que toutes les entrées de cette transaction appartenaient à une seule et même personne (ou entité). Cette heuristique d'analyse on-chain a été découverte par Satoshi Nakamoto lui-même, qui en parle dans la partie 10 du White Paper :

> « *Toutefois, la liaison est inévitable avec les transactions multi-entrées, qui révèlent nécessairement que leurs entrées étaient détenues par un même propriétaire. Le risque est que si le propriétaire d'une clef est révélé, les liaisons peuvent révéler d'autres transactions qui ont appartenu au même propriétaire.* » - Nakamoto, S. (2008). "Bitcoin: A Peer-to-Peer Electronic Cash System". Consulté à l'adresse https://bitcoin.org/bitcoin.pdf.

Encore aujourd'hui, le CIOH demeure la principale heuristique employée par les sociétés d'analyse de chaîne, avec la réutilisation d'adresse.



En français, on pourrait traduire « CIOH » par « Heuristique de propriété commune des entrée ».

CLÉ ÉTENDUE

Suite de caractère qui combine une clé (publique ou privée), son code de chaîne associé et une série de métadonnées. Une clé étendue rassemble en une seule chaîne de caractère tous les éléments nécessaires à la dérivation de clés enfants. Elles sont utilisées dans les portefeuilles déterministes et hiérarchiques, et peuvent être de deux types : une clé publique étendue (utilisée pour dériver des clés publiques enfants) ou une clé privée étendue (utilisée pour dériver à la fois des clés privées et des clés publiques enfants). Une clé étendue inclut donc plusieurs données différentes, décrites au sein du BIP32, dans l'ordre :

- Le préfixe. `prv` et `pub` sont des HRP permettant d'indiquer si l'on a à faire à une clé privée étendue (`prv`) ou à une clé publique étendue (`pub`). La première lettre du préfixe permet, elle, de désigner la version de la clé étendue :
- `x` permet d'indiquer un objectif Legacy ou SegWit V1 sur Bitcoin ;
- `t` permet d'indiquer un objectif Legacy ou SegWit V1 sur Bitcoin Testnet ;
- `y` permet d'indiquer un objectif Nested SegWit sur Bitcoin ;
- `u` permet d'indiquer un objectif Nested SegWit sur Bitcoin Testnet ;
- `z` permet d'indiquer un objectif SegWit V0 sur Bitcoin ;
- `v` permet d'indiquer un objectif SegWit V0 sur Bitcoin Testnet.
- La profondeur, qui indique le nombre de dérivations intervenues depuis la clé maîtresse pour arriver jusqu'à la clé étendue ;
- L'empreinte du parent. Cela représente les 4 premiers octets du `HASH160` de la clé publique parent ;

- L'index. C'est le numéro de la paire parmi ses sœurs dont est issue la clé étendue ;
- Le code de chaîne ;
- Un octet de rembourrage si c'est une clé privée `0x00` ;
- La clé privée ou la clé publique ;
- Une somme de contrôle. Elle incarne les 4 premiers octets du `HASH256` de tout le reste de la clé étendue.

Dans la pratique, la clé publique étendue est utilisée pour générer des adresses de réception et pour observer les transactions d'un compte, sans exposer les clés privées associées. Cela peut permettre, par exemple, la création d'un portefeuille dit « watch-only ». Il est toutefois important de noter que la clé publique étendue est une information sensible pour la confidentialité de l'utilisateur, car sa divulgation peut permettre à des tiers de tracer les transactions et de visualiser le solde du compte associé.

CLÉ PRIVÉE

Une clé privée est un élément fondamental de la cryptographie asymétrique. Il s'agit d'une chaîne de caractères alphanumériques de 256 bits qui représente un secret cryptographique. Cette clé est utilisée pour signer numériquement des transactions et prouver la possession d'une clé publique Bitcoin (et par extension, d'une adresse de réception). Les clés privées permettent donc de dépenser des bitcoins en débloquant les UTXO associés à la clé publique correspondante. Les clés privées doivent être conservées strictement confidentielles, car leur divulgation pourrait permettre à des tiers malveillants de prendre le contrôle des fonds associés. Dans le système Bitcoin, la clé privée est liée à une clé publique par le biais d'un algorithme de signature numérique à courbes elliptiques (ECDSA ou Schnorr). La clé publique est dérivée de la clé privée, mais l'inverse est pratiquement impossible à réaliser en raison de la difficulté computationnelle inhérente à la résolution du problème mathématique sous-jacent (problème du logarithme discret). La clé publique est généralement utilisée pour générer une adresse Bitcoin, qui sert à bloquer des bitcoins à l'aide d'un script. En cryptographie, les clés privées sont souvent des nombres aléatoires ou pseudo-aléatoires. Dans le contexte spécifique des portefeuilles déterministes et hiérarchiques Bitcoin, les clés privées sont dérivées de manière déterministe depuis la graine (seed). Les clés privées sont fréquemment confondues avec la graine (seed) ou avec la phrase de récupération (mnémonique). Pourtant, ces éléments sont bien différents.

> *En anglais, une clé privée se dit « private key ».*

CLÉ PUBLIQUE

La clé publique est un élément essentiel de la cryptographie asymétrique. Elle est générée à partir d'une clé privée en utilisant une fonction mathématique irréversible. Sur Bitcoin, les clés publiques sont dérivées depuis leur clé privée associée grâce aux algorithmes de signature numérique à courbes elliptiques ECDSA ou Schnorr. La clé publique, contrairement à la clé privée, peut être partagée librement sans compromettre la sécurité des fonds. Dans le cadre du protocole Bitcoin, la clé publique sert de base pour créer une adresse Bitcoin, qui est ensuite utilisée pour créer des conditions de dépense sur un UTXO. Les clés publiques sont fréquemment confondues avec la clé maîtresse ou avec les clés étendues (xpub…). Pourtant, ces éléments sont bien différents.

> *En anglais, une clé publique se dit « public key ». Ce terme est parfois abrégé avec « pubkey », ou « PK ».*

CLI

Acronyme de « Command Line Interface », ou « interface en ligne de commande » en français. C'est une méthode d'interaction avec des logiciels qui repose sur la saisie de commandes textuelles dans un terminal ou une console. La CLI se différencie de la GUI (interface graphique utilisateur) qui dispose de méthodes d'interactions de pointage (avec la souris) et d'éléments visuels interactifs.

C-LIGHTNING (CLN)

Ancien nom de l'implémentation Core-Lightning.

> *Pour plus d'informations, voir la définition de **CORE-LIGHTNING**.*

CLONE

Dans le cadre de Git, consiste à créer une copie locale d'un dépôt existant. Cette opération télécharge l'ensemble du dépôt, y compris toutes les branches et l'historique des commits. En tant qu'utilisateur de Bitcoin, il est possible d'avoir à faire à cette commande lorsque l'on télécharge un logiciel.

CODE DE CHAÎNE

Dans le contexte de la dérivation hiérarchique et déterministe (HD) des portefeuilles Bitcoin, le code de chaîne est une valeur de sel cryptographique de 256 bits utilisée pour générer des clés enfants à partir d'une clé parent, selon le standard BIP32. Le code de chaîne est utilisé en combinaison avec la clé parente et l'index de l'enfant pour générer de manière sécurisée et déterministe une nouvelle paire de clés (clé privée et clé publique) sans révéler la clé parente ou les autres clés enfants dérivées. Il existe donc un code de chaîne unique pour chaque paire de clés. Le code de chaîne est obtenu soit en hachant la graine du portefeuille, et en prenant la moitié des bits à droite. Dans ce cas, on parle d'un code de chaîne maître, associé à la clé privée maîtresse. Ou bien, il peut être obtenu en hachant une clé parent avec son code de chaîne parent et un index, et en conservant les bits à droite. On parle alors de code de chaîne enfant. Cette approche permet aux utilisateurs de gérer plusieurs adresses Bitcoin à partir d'une seule graine (seed), améliorant ainsi la confidentialité dans les transactions Bitcoin. Il est impossible de dériver des clés sans avoir la connaissance du code de chaîne associé à chaque paire parent. Il permet d'introduire des données pseudo-aléatoires dans le processus de dérivation pour garantir que la génération des clés cryptographiques reste imprévisible pour les attaquants tout en étant déterministe pour le détenteur du portefeuille.

> *En anglais, un code de chaîne se dit « chain code », et un code de chaîne maître se dit « master chain code ».*

CODE DE PAIMENT RÉUTILISABLE

Dans le BIP47, un code de paiement réutilisable est une information générée à partir d'un portefeuille Bitcoin permettant d'engager une transaction de notification et de dériver des adresses uniques. Cela permet de ne pas faire de réutilisation d'adresses, qui mènent à une perte de la confidentialité, sans pour autant devoir dériver et transmettre manuellement de nouvelles adresses vierges à chaque paiement. Dans le BIP47, les codes de paiement réutilisables sont construits de la manière suivante :

- L'octet 0 correspond à la version ;
- L'octet 1 est un champ de bits permettant d'ajouter des informations en cas d'utilisation spécifique ;

- L'octet 2 permet d'indiquer la parité du `y` de la clé publique ;
- De l'octet 3 à l'octet 34, on retrouvera la valeur `x` de la clé publique ;
- De l'octet 35 à l'octet 66, il y a le code de chaîne associé à la clé publique ;
- De l'octet 67 à l'octet 79, c'est du rembourrage de zéros.

On ajoute généralement un HRP au départ du code de paiement et une somme de contrôle à la fin, puis on l'encode en base58. La construction d'une code de paiement est donc assez proche de celle d'une clé étendue. Voici mon propre code de paiement BIP47 en base58 :

`PM8TJSBiQmNQDwTogMAbyqJe2PE2kQXjtgh88MRTxsrnHC8zpEtJ8j7Aj628oUFk8X6P5rJ7P5qDudE4Hwq9JXSRzGcZJbdJAjM9oVQ1UKU5j2nr7VR5`

Dans l'implémentation PayNym du BIP47, les codes de paiement peuvent également être exprimés sous la forme d'identifiants associés à l'image d'un robot. Voici le mien : `+throbbingpond8B1`. L'utilisation de codes de paiements avec l'implémentation PayNym est actuellement disponible sur Sparrow Wallet sur PC et sur Samourai Wallet sur mobile.

COINBASE (TRANSACTION)

Type spécifique de transaction Bitcoin, unique pour chaque bloc et toujours la première de celui-ci. Elle permet au mineur ayant trouvé une preuve de travail valide de recevoir sa récompense de bloc. Cette récompense se compose de deux éléments : la subvention de bloc, qui génère de nouveaux BTC conformément au calendrier d'émission défini par les règles de consensus, et les frais de transaction, qui correspondent à la différence entre le total des entrées et des sorties de toutes les transactions incluses dans le bloc. La particularité de la transaction Coinbase est qu'elle est la seule à ne pas requérir d'entrée (input), ce qui signifie qu'elle crée des bitcoins ex nihilo. Elle inclut également parfois des informations de gestion choisies par le mineur ou la pool de minage, telles que des messages ou des données sur la version du logiciel utilisé. Les bitcoins générés par une transaction Coinbase sont soumis à une période de maturité de 100 blocs pendant laquelle ils ne peuvent pas être dépensés par le mineur.

> *Il n'existe aucune traduction de « Coinbase » en français. Il est donc admis d'utiliser directement ce terme.*

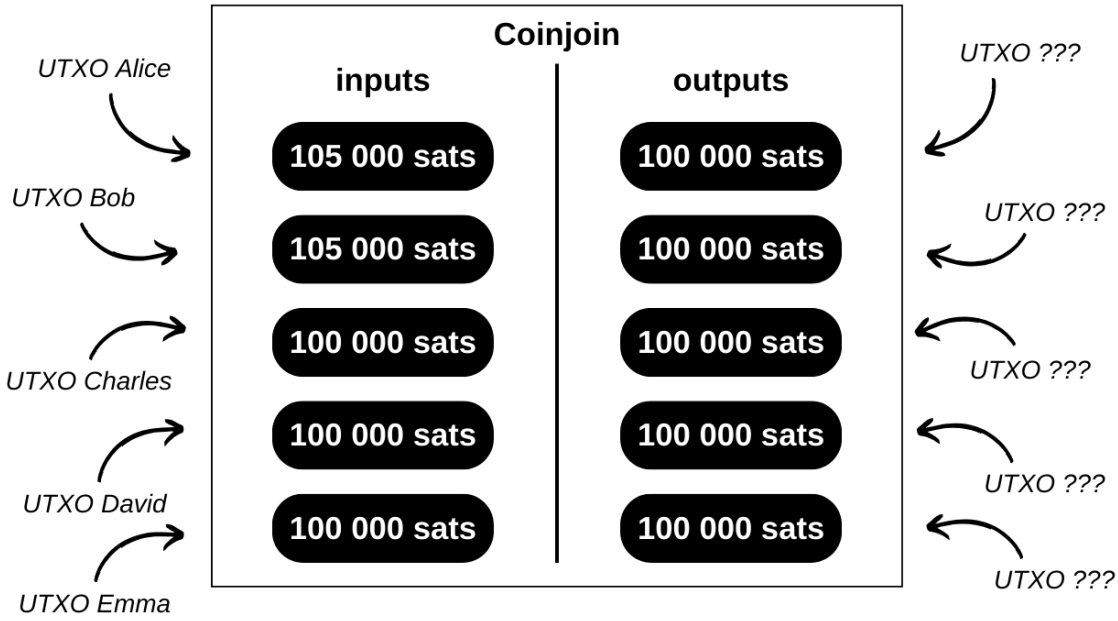
COIN CONTROL

Fonctionnalité présente dans certains logiciels de portefeuille Bitcoin, qui donne aux utilisateurs la capacité de sélectionner manuellement les UTXO spécifiques à utiliser en tant qu'entrées pour effectuer une transaction. En d'autres termes, le coin control offre la possibilité de choisir précisément quels morceaux de bitcoins seront dépensés. Cette fonctionnalité est similaire à l'action de choisir une pièce spécifique de votre porte-monnaie pour payer votre baguette. Le coin control est particulièrement utile pour gérer ses frais de transaction et pour améliorer sa confidentialité. En effet, en sélectionnant spécifiquement les UTXO à utiliser, les utilisateurs peuvent éviter de fusionner des UTXO issus de sources différentes, ce qui pourrait révéler des informations sur l'ensemble de leurs fonds (CIOH). Cette fonctionnalité va souvent de pair avec la possibilité d'étiqueter les sorties de transaction.

COINJOIN

Le Coinjoin est une technique permettant de casser le traçage des bitcoins. Il repose sur une transaction collaborative à la structure spécifique de même nom : la transaction Coinjoin. Les transactions

Coinjoin permettent d'améliorer la protection de la vie privée des utilisateurs de Bitcoin en rendant l'analyse des transactions plus difficile pour les observateurs extérieurs. Cette structure permet la combinaison de plusieurs transactions indépendantes en une seule transaction, rendant difficile la détermination des liens entre les adresses d'entrée et de sortie. Le fonctionnement général du Coinjoin est le suivant : différents utilisateurs souhaitant mixer déposent un montant en input d'une transaction. Ces inputs ressortiront en différents outputs de même montant. À la sortie de la transaction, il est donc impossible de déterminer quel output appartient à quel utilisateur. Il n'y a techniquement aucun lien entre les entrées et les sorties de la transaction Coinjoin. Le lien entre chaque utilisateur et chaque UTXO est cassé, de la même manière que l'historique de chaque pièce.



Pour permettre le Coinjoin sans qu'aucun utilisateur ne perde la main sur ses fonds à aucun moment, la transaction est d'abord construite par un coordinateur puis transmise à chaque utilisateur. Chacun d'eux signe alors la transaction de son côté en vérifiant qu'elle lui convient, puis toutes les signatures sont ajoutées à la transaction. Si un utilisateur ou le coordinateur tente de voler les fonds des autres en modifiant les outputs de la transaction Coinjoin, alors les signatures seront invalides et la transaction sera refusée par les nœuds. Ce protocole spécifique avec un coordinateur central s'appelle « *Chaumian Coinjoin* ».

Ce mécanisme augmente la confidentialité des transactions sans nécessiter de modifications du protocole Bitcoin. Des implémentations spécifiques de Coinjoin, telles que Whirlpool, JoinMarket ou Wabisabi, proposent des solutions pour faciliter le processus de coordination entre les participants et renforcer l'efficacité de la transaction Coinjoin. Exemple de transaction Coinjoin : 323df21f0b0756f98336437aa3d2fb87e02b59f1946b714a7b09df04d429dec2

> *Le terme de « Coinjoin » ne dispose pas de traduction française. Certains bitcoiners utilisent également les termes de « mix », de « mixing » ou encore de « mixage » pour évoquer la transaction Coinjoin. Le mixage est plutôt le processus utilisé au cœur du Coinjoin. Aussi, il ne faut pas confondre le mixage par Coinjoins et le mixage par un acteur central qui prend possession des bitcoins durant le processus. Cela n'a rien à voir avec le Coinjoin où l'utilisateur ne perd à aucun moment la main sur ses bitcoins durant le processus. Pour plus d'informations, voir la définition de* ***CHAUMIAN COINJOIN****.*

COINJUMBLE

Logiciel développé par Chris Belcher et lancée en août 2014 conçu pour faciliter l'utilisation de coinjoins avec une GUI. À la différence d'autres implémentations de coinjoins de l'époque nécessitant que les participants souhaitent effectuer un coinjoin simultanément, CoinJumble permettait de partager les parties de transaction de manière asynchrone. Les utilisateurs pouvaient communiquer via des canaux de communications externes pour échanger ces parties de transaction encodées. Aujourd'hui, CoinJumble n'est plus utilisé.

COINMUX

Implémentation de coinjoin développée en 2014. Coinmux est un protocole de mixage de bitcoins qui repose sur la confiance réciproque entre les participants, sans nécessiter l'intervention d'un tiers de confiance. Le logiciel regroupe les bitcoins de plusieurs utilisateurs dans une transaction unique, où chaque sortie dispose de montants identiques, ce qui permet de casser les liens entre les ins et les outputs. Le protocole de Coinmux assure que les utilisateurs conservent le contrôle de leurs fonds durant tout le processus, en ne faisant signer les transactions que lorsque les entrées et les sorties correspondent exactement à ce qui a été convenu.

*Pour plus d'informations, voir la définition de **COINJOIN**.*

COINS/

Nom de l'ancien dossier utilisé dans Bitcoin Core pour stocker l'UTXO set remplacé par le fichier chainstate/ dans la version 0.8.0.

*Pour plus d'informations, voir la définition de **UTXO SET**.*

COINSHUFFLE

Protocole de mixage de pièces bitcoins proposé en 2014 par Tim Ruffing, Pedro Moreno-Sanchez, and Aniket Kate, inspiré de l'idée du Coinjoin de Gregory Maxwell. Coinshuffle permet de couper l'historique de pièces sans nécessiter de tiers de confiance. Le protocole assure que même l'intermédiaire ne peut pas relier le payeur au bénéficiaire. Ce concept n'a jamais été largement adopté, les techniques de confidentialité telles que le Chaumian Coinjoin lui étant préférées.

*Pour plus d'informations, voir la définition de **COINJOIN**.*

COLD WALLET

Synonyme de « hardware wallet ».

*Pour plus d'informations, voir la définition de **HARDWARE WALLET**.*

COMMERÇANT

Toute personne physique ou morale qui accepte d'échanger un bien ou un service contre des bitcoins. Ce sont ces commerçants qui confèrent son utilité à la monnaie bitcoin. Plus une monnaie est acceptée par un large éventail de commerçants, plus elle devient utile pour les individus. Puisque les commerçants ont la capacité de déterminer l'utilité d'une monnaie en acceptant de l'échanger contre des biens et des services, dans le cas de Bitcoin, ils ont également un poids considérable dans le choix des règles de consensus. Chacun dispose d'un certain pouvoir proportionnel à l'activité

économique qu'il est en capacité d'apporter à un fork. Parmi les commerçants, il y a évidemment les commerce, mais aussi les plateformes d'échange, les mineurs et les utilisateurs.

COMMIT

Dans le cadre de Git, représente une capture instantanée des modifications apportées à l'ensemble de fichiers d'un dépôt. Chaque commit est identifié par un hachage unique et inclut un message descriptif, l'identité de l'auteur et la date. Il permet de suivre l'évolution du projet et de revenir à des états antérieurs si nécessaire.

COMPACT BLOCK RELAY

Protocole introduit dans Bitcoin Core en 2016 via le BIP152 qui propose une méthode d'économie de bande passante pour les nœuds du réseau. Compact Block Relay permet de communiquer les informations des blocs de manière compacte, en se basant sur l'hypothèse que les nœuds ont déjà une grande partie des transactions d'un bloc récent dans leur mempool. Plutôt que de transmettre chaque transaction intégralement, ce qui constituerait un doublon, Compact Block Relay propose d'envoyer uniquement de courts identifiants pour les transactions déjà connues des pairs, accompagnés de quelques transactions sélectionnées (notamment la transaction coinbase et celles que le nœud est susceptible de ne pas connaître). Le nœud peut ensuite demander à ses pairs les éventuelles transactions manquantes. Compact Block Relay permet ainsi de diminuer la quantité de données échangées lors de la propagation des blocs, ce qui réduit ainsi les pics de bande passante et améliore l'efficacité globale du réseau.

COMPATIBILITÉ RÉTROSPECTIVE

Dans le contexte de Bitcoin, fait référence à la capacité d'une mise à jour des règles du protocole à maintenir la compatibilité avec les versions antérieures. Cela signifie que les modifications sont conçues de manière à ce que les anciens nœuds (les nœuds exécutant des versions antérieures au changement de règles) puissent toujours interagir avec le réseau et suivre la chaîne avec le plus de preuve travail accumulée. Il faut donc que les anciens nœuds ne rejettent ni les nouveaux blocs, ni les nouvelles transactions. La compatibilité rétrospective permet de réduire fortement la probabilité qu'une mise à jour fragmente le réseau, évitant ainsi la division du réseau en sous-groupes sur des chaînes différentes. Pour assurer une compatibilité avec les versions antérieures du protocole, une mise à jour doit rendre les règles existantes plus strictes ou en introduire de nouvelles. C'est ce principe qui définit un « soft fork ». À l'inverse, si une mise à jour assouplit les règles existantes ou en élimine certaines, alors elle ne sera pas rétrocompatible. Ce sera donc un « hard fork ».

CONCATÉNATION

La concaténation, dans le contexte de la cryptographie et des systèmes informatiques, désigne le processus d'assemblage de deux opérandes, en les mettant bout à bout, formant ainsi une nouvelle chaîne de caractères ou de données. Cette opération se note généralement avec un symbole de deux barres verticales $\|$, ou avec le symbôle $\circ$. Par exemple, la concaténation de 45 avec 87 sera égale à 4587. Nous noterons : $45 \| 87 = 4587$. On a mis bout à bout les deux opérandes.

CONDENSAT (HASH)

Le condensat, dans le contexte de la cryptographie, désigne le résultat (ou l'output) produit par l'application d'une fonction de hachage cryptographique à un ensemble de données. Le condensat est une chaîne de caractères de taille fixe généralement représentée sous forme d'une série de

chiffres et de lettres en notation hexadécimale (base 16). Ce résultat a la particularité d'être presque unique et spécifique aux données d'entrée, de sorte qu'un changement minime dans l'entrée produira un condensat complètement différent. Les fonctions de hachage cryptographiques sont conçues pour être unidirectionnelles et résistantes aux collisions, rendant très difficile de retrouver les données initiales à partir du condensat ou de trouver deux entrées distinctes produisant le même condensat.

*Pour plus d'informations, voir la définition de **FONCTION DE HACHAGE**.*

CONFIRMATION

Correspond au nombre de blocs dont pour lesquels une transaction bénéficie de leurs sécurité. Lorsque l'on diffuse une transaction au réseau Bitcoin, celle-ci est d'abord en attente dans les mempools des nœuds. Elle est ensuite incluse dans un bloc valide par un mineur. À ce stade, la transaction vient d'être ajoutée à la blockchain, elle bénéficie donc d'une première confirmation. Lorsqu'un nouveau bloc sera trouvé par dessus le bloc où se trouve la transaction en question, elle bénéficiera d'une seconde confirmation, et ainsi de suite. Chaque nouveau bloc miné par dessus le bloc contenant la transaction constitue une nouvelle confirmation. Grâce au comptage du nombre de confirmations pour une transaction, on peut estimer le risque qu'elle puisse être finalement annulée. Le nombre de confirmation nous permet de juger du niveau d'immuabilité d'une transaction sur la blockchain.

CONSENSUS

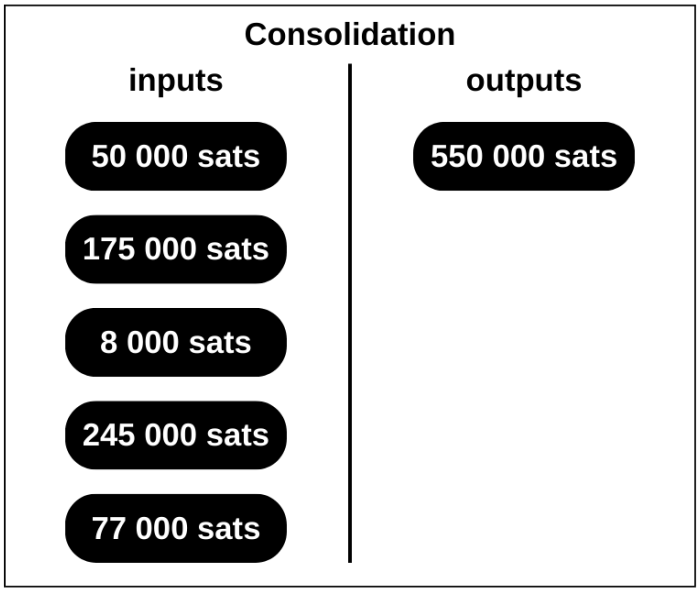
Mécanisme par lequel tous les nœuds du réseau Bitcoin parviennent à s'accorder sur l'état partagé de la blockchain. Le consensus permet que tous les utilisateurs s'alignent sur un même historique des transactions Bitcoin, afin notamment d'éviter la double dépense. Le mécanisme de consensus de Bitcoin est parfois appelé « Consensus de Nakamoto ». Il s'appuie sur la preuve de travail et spécifie que tous les nœuds du réseau acceptent la chaîne disposant de la plus grande quantité de travail accumulé.

Par extension, certaines personnes appellent par « Consensus » les règles tacites du protocole Bitcoin.

CONSOLIDATION

Transaction spécifique dans laquelle plusieurs petits UTXO sont fusionnés en entrée pour former un seul et plus gros UTXO en sortie. Cette opération est une transaction effectuée vers son propre portefeuille. L'objectif de la consolidation est de tirer profit des périodes où les frais sur le réseau Bitcoin sont bas pour fusionner plusieurs petits UTXO en un seul plus grand en valeur. Ainsi, on anticipe les dépenses obligatoires en cas de hausse des frais, permettant d'économiser sur les frais de transaction futurs. En effet, les transactions comportant de nombreuses entrées sont plus lourdes et, par conséquent, plus coûteuses. Outre l'économie réalisable sur les frais de transaction, la consolidation est aussi une forme de planification à long terme. Si votre portefeuille contient de très petits UTXO, ceux-ci peuvent devenir inutilisables si le réseau Bitcoin entre dans une période prolongée de frais élevés. Par exemple, si vous devez dépenser un UTXO de 10 000 satoshis mais que les frais de minage minimums s'élèvent à 15 000 satoshis, la dépense excéderait la valeur de l'UTXO lui-même. Ces petits UTXO deviennent alors économiquement non rationnels à utiliser et restent inutilisables tant que les frais ne baissent pas. Ces UTXO sont communément appelés « dust » (poussière). En consolidant régulièrement vos petits UTXOs, vous réduisez ce risque associé aux augmentations de frais. Cependant, il est important de noter que les transactions de consolidation sont reconnaissables lors d'une analyse de chaîne. Une telle transaction indique une CIOH (*Common Input Ownership Heuristic*), c'est-à-dire que les entrées de la transaction de consolidation sont

possédées par une seule entité. Cela peut avoir des implications en termes de confidentialité pour l'utilisateur.



CONTRAT INTELLIGENT

Programme qui s'exécute automatiquement lorsque certaines conditions prédéfinies sont remplies. Un contrat intelligent est donc un ensemble de clauses entre plusieurs parties qui peuvent se réaliser sans nécessiter l'intervention d'un tiers de confiance. Ces contrats déclenchent généralement des actions spécifiques comme un transfert de bitcoins.

> *En anglais, on parle de « Smart Contract ». En français, on parle également parfois de « contrat autonome ».*

COOKIE (.COOKIE)

Fichier utilisé pour l'authentification RPC (*Remote Procedure Call*) dans Bitcoin Core. Lorsque bitcoind démarre, il génère un cookie d'authentification unique et le stocke dans ce fichier. Les clients ou les scripts qui souhaitent interagir avec bitcoind via l'interface RPC peuvent utiliser ce cookie pour s'authentifier de manière sécurisée. Ce mécanisme permet une communication sûre entre le bitcoind et les applications externes, sans nécessiter une gestion manuelle des noms d'utilisateur et des mots de passe. Le fichier .cookie est régénéré à chaque redémarrage de bitcoind et supprimé à l'arrêt.

CORE-LIGHTNING (CLN)

Implémentation majeure du protocole Lightning Network écrite en langage C et Rust. Développée par Blockstream, Core-Lightning est conçue pour être légère et performante. Elle se distingue par son architecture modulaire, permettant aux développeurs d'ajouter facilement des fonctionnalités personnalisées. Cette implémentation a été renommée en 2022. Son nom original était auparavant « C-Lightning ».

COURBE ELLIPTIQUE

Dans le contexte de la cryptographie, une courbe elliptique est une courbe algébrique définie par une équation de la forme $y^2 = x^3 + ax + b$. Ces courbes sont utilisées dans la cryptographie à courbes elliptiques (ECC), qui est une méthode de cryptographie à clé publique permettant de créer des algorithmes de chiffrement, de signature numérique et d'échange de clés. Dans le contexte de Bitcoin, l'algorithme ECDSA (Elliptic Curve Digital Signature Algorithm) ou le protocole de Schnorr sont utilisés avec la courbe `secp256k1`. Cette courbe a été choisie pour ses propriétés de performance et de sécurité. Ces algorithmes sont utilisés pour générer des clés publiques à partir de clés privées, ainsi que pour signer des transactions, et donc débloquer des bitcoins.

COVENANT

Mécanisme qui permet d'imposer des conditions spécifiques sur la manière dont une pièce donnée peut être dépensée, y compris dans des transactions futures. Au-delà des conditions usuellement autorisées par le langage script sur un UTXO, le covenant force des contraintes supplémentaires sur la manière de dépenser cette pièce Bitcoin dans des transactions ultérieures. Techniquement, l'instauration d'un covenant intervient lorsque le `scriptPubKey` d'un UTXO définit des restrictions sur le `scriptPubKey` des sorties d'une transaction qui dépense ledit UTXO. En élargissant la portée de script, les covenants permettraient de nombreuses évolutions sur Bitcoin comme l'ancrage bilatéral des drivechains, la mise en place de vaults ou encore l'amélioration des systèmes de surcouche comme Lightning. On différencie les propositions de covenants en fonction de trois critères :

- Leur porté ;
- Leur implémentation ;
- Leur récursivité.

Il existe de très nombreuses propositions qui permettraient l'utilisation de covenants sur Bitcoin. Les plus avancées dans le processus d'implémentation sont : OP_CHECKTEMPLATEVERIFY (CTV), SIGHASH_ANYPREVOUT (APO) et OP_VAULT. Parmi les autres propositions, il y a : OP_TX, OP_TAPLEAFUPDATEVERIFY (TLUV), OP_EVICT, OP_CHECKSIGFROMSTACKVERIFY, etc.

Pour bien comprendre le concept de covenant, je vous propose une analogie : imaginez un coffre-fort contenant 500 € en petites coupures. Si vous parvenez à déverrouiller ce coffre avec la clé adéquate, alors vous êtes libre d'utiliser cet argent comme bon vous semble. Ça, c'est la situation normale de Bitcoin. Maintenant, imaginez que ce même coffre-fort ne contient pas 500 € en billets de banque, mais plutôt des tickets restaurants d'une valeur équivalente. Si vous réussissez à ouvrir ce coffre, vous pouvez disposer de cette somme. Cependant, votre liberté de dépense est restreinte :vous ne pouvez utiliser ces tickets pour payer que dans certains restaurants. Ainsi, il y a une première condition pour dépenser cet argent, qui est de parvenir à ouvrir le coffre avec la clé appropriée. Mais il y a aussi une condition supplémentaire quant à l'usage futur de cette somme : elle doit être dépensée exclusivement dans des restaurants partenaires, et non pas en toute liberté. Ce système de contraintes sur les transactions futures, c'est ce que l'on appelle un covenant.

> *En français, il n'existe aucun terme pour capturer précisément la signification du mot « covenant ». On pourrait parler de « clause », de « pacte » ou d' « engagement », mais cela ne correspondrait pas exactement au terme « covenant ». Ce dernier est d'ailleurs emprunté d'une terminologie juridique qui permet de décrire une clause contractuelle imposant des obligations persistantes sur un bien.*

CPFP (CHILD PAY FOR PARENT)

Mécanisme transactionnel visant à accélérer la confirmation d'une transaction Bitcoin, tout comme le fait Replace-by-Fee (RBF), mais du côté du destinataire. Lorsqu'une transaction avec des frais trop faibles par rapport au marché reste bloquée dans les mempools des nœuds et ne se confirme pas assez rapidement, le destinataire peut initier une nouvelle transaction, dépensant les bitcoins reçus dans la transaction bloquée, bien qu'elle ne soit pas encore confirmée. Cette seconde transaction nécessite forcément que la première soit minée pour être confirmée. Les mineurs sont donc obligés d'inclure les deux transactions ensemble. La seconde va allouer beaucoup plus de frais de transaction que la première, de telle sorte que la moyenne de frais incite les mineurs à inclure les deux transactions. La transaction enfant (la seconde) paie pour la transaction parent qui est bloquée (la première). C'est pour cela que l'on parle d'un « CPFP ». Ainsi, CPFP permet au destinataire d'obtenir plus rapidement ses fonds malgré les faibles frais initiaux engagés par l'expéditeur.

CPPSRB

Sigle de « *Capped Pay Per Share Recent Backpay* ». C'est une méthode de calcul de la rémunération des mineurs dans le contexte des pools de minage. Dans ce système, la pool paie autant de parts (shares) qu'elle le peut à chaque fois qu'un bloc est trouvé, en donnant la priorité aux parts les plus récentes. Cette méthode garantit que la probabilité de trouver un bloc reste constante, indépendamment de la durée du cycle de minage en cours, offrant ainsi une certaine protection contre le pool hopping.

CPU (CENTRAL PROCESSING UNIT)

Composant principal d'un ordinateur responsable de l'exécution des instructions machines des logiciels. Dans le contexte de Bitcoin, le CPU était initialement utilisé pour le minage par les nœuds avant d'être surpassé par le minage par GPU (cartes graphiques), puis par l'utilisation de puces spécialisées que l'on appelle des « ASIC ».

> *En français, on peut parler d'une « unité centrale de calcul » ou bien simplement d'un « processeur ».*

CRYPTANALYSE

Étude des techniques mathématiques pour tenter de casser les techniques cryptographiques. Cela inclut les processus de recherche d'erreurs ou de faiblesses dans l'implémentation d'une méthode cryptographique ou dans la méthode cryptographique elle-même.

CRYPTER

Ce terme n'existe pas. On dit « chiffrer ».

CRYPTO-ACTIF

Terme utilisé dans un contexte juridique et réglementaire pour désigner les divers types de cryptomonnaies, dont le bitcoin.

CRYPTOGRAPHIE

Discipline qui incarne les principes, les moyens et les méthodes de transformation des informations, notamment avec des techniques mathématiques, afin de masquer leur contenu sémantique,

d'empêcher leur utilisation non autorisée, d'assurer leur authenticité ou d'empêcher leur modification non détectée. Elle regroupe l'utilisation d'algorithmes de hachage, de signature numérique et de chiffrement.

CRYPTOLOGIE

Science mathématique qui traite de la cryptanalyse et de la cryptographie.

CRYPTOMONNAIE

Qualificatif générique donné à toute forme de monnaie, d'actif, de crédit ou d'unité numérique au sein d'un système informatique dans lequel on utilise de la cryptographie pour les échanges et les transactions entre les utilisateurs.

CYPHERPUNK

Communauté informelle et internationale de personnes intéressées par l'utilisation de la cryptographie comme moyen pour assurer les libertés individuelles. Les cypherpunks prônent l'utilisation de la cryptographie pour imposer son droit fondamental de protéger sa vie privée en tant qu'individu, en particulier dans un contexte d'augmentation de la surveillance gouvernementale et de l'exploitation des données par des entités privées. L'histoire des cypherpunks remonte aux années 1980 et 1990, lorsque des groupes de cryptographes, de programmeurs et de libertaires commencent à discuter et à promouvoir l'utilisation de la cryptographie pour protéger l'anonymat et la liberté individuelle. Un moment clé est la fondation, en 1992, de la « Cypherpunks mailing list », une liste de diffusion par courrier électronique qui a servi pour ces discussions. La publication en 1993 du *Cypherpunk's Manifesto* par Eric Hughes a également été un moment important. Ce document décris les objectifs et les actions des cypherpunks. L'idée d'une monnaie électronique qui ne s'établit pas sur une entité centrale, comme Bitcoin, est enracinée dans la philosophie cypherpunk. La création de Bitcoin est souvent considérée comme une réalisation majeure de cette vision.

D

DANDELION

Proposition qui vise à améliorer la confidentialité du routage des transactions dans le réseau Bitcoin pour contrer la désanonymisation. Dans le fonctionnement classique de Bitcoin, les transactions sont immédiatement diffusées à de multiples nœuds. Ce phénomène peut potentiellement permettre à des observateurs de lier des transactions, normalement anonymes, avec des adresses IP. L'objectif du BIP156 est de traiter ce problème. Pour ce faire, il introduit une phase supplémentaire dans la diffusion permettant de préserver l'anonymat avant la propagation publique. Ainsi, Dandelion utilise d'abord une phase de « tige » où la transaction est envoyée à travers un chemin aléatoire de nœuds, avant d'être diffusée à l'ensemble du réseau dans la phase de « capitule ». La tige et le capitule sont des références au comportement de la propagation de la transaction à travers le réseau, qui ressemble à la forme d'un pissenlit (« *a dandelion* » en anglais). Cette méthode de routage brouille la piste menant au nœud source, rendant difficile de retracer une transaction via le réseau jusqu'à son origine.

*Pour plus d'informations, voir la définition de **BIP156**.*

DARKWALLET

Logiciel de portefeuille Bitcoin axé sur la confidentialité, lancé par Amir Taaki et Cody Wilson en 2014, fonctionnant comme une extension pour le navigateur Google Chrome. DarkWallet disposait de fonctionnalités pour améliorer la confidentialité de l'utilisateur de Bitcoin, telles que les paiements furtifs et les coinjoins. Son développement a été abandonné depuis janvier 2015.

DATABASE/

Ancien dossier contenant des bases de données pour le portefeuille Bitcoin Core. Depuis la version 0.16, cette base de données a été déplacée dans le dossier wallet/.

DB.LOG

Ancien fichier log (historique des événements) du portefeuille Bitcoin Core déplacé dans le dossier wallet/ depuis la version 0.16.

DDOS

Forme de DOS où l'attaque provient de multiples sources simultanément, rendant la défense plus complexe. Les attaquants utilisent souvent des réseaux d'ordinateurs infectés par des virus (botnets) pour lancer des requêtes massives vers une seule cible. Cette stratégie multiplie l'efficacité de l'attaque en surchargeant les capacités du système ciblé afin de provoquer des interruptions de service.

En français, on peut le traduire par « attaque par déni de service distribué ».

DEBUG.LOG

Fichier contenant l'historique des événements de Bitcoin Core. Il contient des données de journalisation, telles que les messages d'erreur, les avertissements et d'autres informations de débogage. Ce fichier est utilisé pour résoudre des éventuels problèmes techniques.

DÉPÔT

Structure de données centrale utilisée dans Git où sont stockées les informations de versionnage d'un projet. Un dépôt contient l'historique complet de toutes les modifications, les branches et les tags. Chaque dépôt est une collection indépendante de fichiers et de dossiers, accompagnée d'un historique des commits, permettant la collaboration et le suivi des changements au fil du temps. Par exemple, le dépôt de Bitcoin Core est stocké sur Github ici : https://github.com/bitcoin/bitcoin.

> *En anglais, on parle d'un « repository ». Il est courant d'employer la troncation « repo » pour désigner un dépôt Git.*

DGM

Sigle de « *Double Geometric Method* ». C'est une méthode de calcul de la rémunération des mineurs dans le contexte des pools de minage. C'est une méthodes hybride, qui est sensée combiner les avantages de PPLNS et de la méthode dite « géométrique ». Elle dispose d'une faible variance sur les parts, à la manière de PPLNS, puis permet au mineur d'absorber de la variance pour réduire celle de la pool dans un second temps. DGM est résistant au pool hopping en garantissant que le paiement attendu par part reste constant. La méthode est basée sur des scores, rendant les paiements indépendants de l'historique de la pool et presque totalement indépendants des changements futurs de difficulté.

DIFFIE-HELLMAN

Méthode cryptographique permettant à deux parties de générer un secret partagé sur un canal de communication non sécurisé. Ce secret peut ensuite servir à chiffrer la communication entre les deux parties. Diffie-Hellman utilise l'arithmétique modulaire pour que, même si un attaquant peut observer les échanges, il ne peut pas déduire le secret partagé sans résoudre un problème mathématique difficile : le logarithme discret.

DISTRIBUÉ

Attribut d'un réseau informatique dans lequel le pouvoir de décision et le contrôle sont répartis de manière équitable entre tous les participants du réseau. Cette répartition garantit la résilience du système. On parle également de réseau pair-à-pair. Contrairement à un réseau décentralisé, où le pouvoir est fragmenté et dispersé parmi plusieurs entités, mais où certaines autorités centrales demeurent dotées d'un pouvoir supérieur à celui des utilisateurs, un réseau distribué élimine l'autorité centrale en confiant la gestion et le contrôle aux utilisateurs eux-mêmes. Bitcoin est un exemple de réseau distribué. Comme protocole de cash électronique pair-à-pair, Bitcoin se distingue par son absence de hiérarchie et d'autorité centrale. La tenue du consensus, la vérification des transactions et l'émission de nouvelles unités monétaires sont réalisées par les utilisateurs du réseau. Cette structure distribuée assure la résilience et la résistance à la censure du système, rendant très difficile pour une entité unique de contrôler ou de manipuler le réseau.

> *Certaines personnes parlent de Bitcoin comme d'un système décentralisé. En effet, il n'est pas rare d'observer une interchangeabilité de ces deux termes. Un synonyme plus évocateur de l'adjectif « distribué » pourrait être « pair-à-pair », parfois abrégé « P2P », le sigle de la traduction anglaise « Peer-to-Peer ».*

DLC (DISCREET LOG CONTRACT)

Type de contrat intelligent sur Bitcoin qui permet l'exécution de conditions contractuelles à partir du résultat d'événements externes, validés par un ou plusieurs oracles, sans que ces derniers ne connaissent les détails du contrat. Les DLC ont été inventés par Tadge Dryja en 2018. Ces contrats intelligents sont principalement utiles dans des applications financières, permettant par exemple de créer des instruments financiers ou des paris conditionnels, tout en réduisant les risques de contrepartie. Pour construire un DLC, plusieurs parties bloquent des bitcoins sur une adresse multisig. Ces bitcoins ne peuvent être débloqués que lorsque l'oracle publie les informations spécifiées à un moment donné.

DLP (DISCREET LOG PROBLEME)

Voir la définition de **LOGARITHME DISCRET (PROBLÈME)**.

DNS SEEDS

Points de connexion initiaux pour les nouveaux nœuds Bitcoin qui rejoignent le réseau. Ces seeds, qui sont en fait des serveurs DNS spécifiques, ont leur adresse intégrée de façon permanente dans le code de Bitcoin Core. Lorsqu'un nouveau nœud se lance, il contacte ces serveurs pour obtenir une liste aléatoire d'adresses IP de nœuds Bitcoin à priori actifs. Le nouveau nœud pourra ainsi établir des connexions avec les nœuds de cette liste afin d'obtenir les informations pour faire son IBD et se synchroniser sur la chaîne avec le plus de travail accumulé. En 2023, voici la liste des DNS seeds de Bitcoin Core et les personnes responsables de leur maintenance (bitcoin/src/kernel/chainparams.cpp) :

- seed.bitcoin.sipa.be : Pieter Wuille ;
- dnsseed.bluematt.me : Matt Corallo ;
- dnsseed.bitcoin.dashjr.org : Luke Dashjr ;
- seed.bitcoinstats.com : Christian Decker ;
- seed.bitcoin.jonasschnelli.ch : Jonas Schnelli ;
- seed.btc.petertodd.net : Peter Todd ;
- seed.bitcoin.sprovoost.nl : Sjors Provoost ;
- dnsseed.emzy.de : Stephan Oeste ;
- seed.bitcoin.wiz.biz : Jason Maurice.

Les DNS seeds représentent le second moyen, par ordre de priorité, pour un nœud de Bitcoin d'établir des connexions. Le premier moyen consiste à utiliser le fichier peers.dat que le nœud a lui-même créé. Ce fichier est naturellement vide dans le cas d'un nouveau nœud, à moins que l'utilisateur l'ai modifié manuellement.

> *Attention, les DNS seeds ne doivent pas être confondus avec les « seed nodes », qui sont eux la troisième manière d'établir des connexions. Pour plus d'informations, voir la définition de* ***SEED NODES****.*

DOS (DENIAL OF SERVICE)

Attaque informatique qui vise à rendre une ressource (site web, nœud, service en ligne…) indisponible pour ses utilisateurs légitimes. Les attaquants surchargent la cible avec un volume de données ou de

requêtes excessivement élevé, ce qui épuise les ressources système et réseau de la victime, entraînant des ralentissements ou un arrêt complet. Les méthodes de DoS peuvent varier, mais l'objectif reste le même : empêcher l'accès à des services ou des données. Dans le contexte spécifique de Bitcoin, une attaque DoS peut viser à saturer le réseau ou les nœuds avec un volume excessif de requêtes afin d'entraver leur fonctionnement normal. L'objectif est souvent de nuire à un opérateur de nœud ou à la disponibilité du réseau pour les utilisateurs honnêtes.

En français, on peut le traduire par « attaque par déni de service ».

DOUBLE DÉPENSE (ATTAQUE)

Attaque où un utilisateur malveillant tente d'utiliser le même UTXO (*Unspent Transaction Output*) plus d'une fois afin de s'enrichir sur les contreparties des transactions impliquées. En principe, une fois qu'une transaction est confirmée dans un bloc et ajoutée à la blockchain, l'utilisation de ces bitcoins est enregistrée de manière permanente, empêchant toute dépense ultérieure de ces mêmes bitcoins. Prévenir la double dépense est même l'utilité première de la blockchain. Dans le cadre d'une attaque de double dépense, l'attaquant effectue d'abord une transaction légitime auprès d'un commerçant, puis crée une seconde transaction concurrente qui dépense les mêmes pièces, soit en les renvoyant vers lui-même pour récupérer la somme, soit en les utilisant pour acheter un autre bien ou service auprès d'un autre commerçant. Deux scénarios principaux peuvent permettre cette attaque. Le premier, et le plus simple pour l'attaquant, consiste à exécuter la transaction frauduleuse avant que la transaction légitime ne soit incluse dans un bloc. Pour permettre la confirmation de sa transaction frauduleuse en première, l'attaquant y associe des frais de transaction nettement plus élevés que la transaction légitime. C'est une sorte de « RBF » frauduleux. Ce scénario n'est possible que si le commerçant accepte de finaliser la vente en « zeroconf », c'est-à-dire sans aucune confirmation pour la transaction de paiement. C'est pourquoi il est fortement recommandé d'attendre plusieurs confirmations avant de considérer une transaction comme immuable. Le second scénario, beaucoup plus complexe, est celui d'une attaque à 51 %. Si l'attaquant contrôle une part importante de la puissance de calcul du réseau, il peut miner une chaîne concurrente à celle contenant la transaction légitime, mais incluant sa transaction frauduleuse. Lorsque le commerçant a accepté la vente et que l'attaquant a réussi à créer une chaîne plus longue (avec plus de travail accumulé) que la chaîne légitime, il peut alors diffuser sa chaîne frauduleuse qui sera reconnue par les nœuds du réseau comme étant celle valide.

DRIVECHAIN

Forme spécifique de sidechain où les mineurs de la blockchain principale (Bitcoin) ont un rôle direct dans la gouvernance de l'ancrage bilatéral et éventuellement dans le mécanisme de consensus de la sidechain. Ce protocole a été inventé par Paul Sztorc et pourrait être mis en place grâce aux controversés BIP300, qui permettrait le two-way peg auprès des mineurs, et BIP301, qui permettrait d'utiliser le minage fusionné (merged mining) de manière optimale.

DUMMY ELEMENT

Fait référence à un élément supplémentaire et inutile consommé par les opcodes `OP_CHECKMULTISIG` et `OP_CHECKMULTISIGVERIFY` lors de la vérification des signatures dans une transaction. En raison d'un bug off-by-one historique (erreur de décalage unitaire), ces 2 opcodes suppriment un élément supplémentaire sur la pile en plus de leur fonction de base. Pour éviter une erreur, il est donc obligatoire d'inclure une valeur factice au début du `ScriptSig` afin de satisfaire la suppression et outrepasser le bug. Cette valeur inutile, c'est ce que l'on appelle le « *dummy element* ». Le BIP11, qui a introduit le standard P2MS, conseillait de mettre un `OP_0` comme valeur inutile. Mais ce standard

n'était pas imposé au niveau des règles de consensus, ce qui veut dire que n'importe quelle valeur pouvait y être placée, sans invalider la transaction. Le dummy element était donc un vecteur de malléabilité des transactions. Le BIP147, introduit avec le soft fork SegWit, a imposé que cet élément factice soit strictement un tableau d'octets vide (`OP_0`), éliminant ainsi la malléabilité associée à cet élément en rendant toute transaction non conforme invalide selon les règles de consensus. Cette règle, nommée `NULLDUMMY`, s'applique à la fois aux transactions SegWit et pré-SegWit.

*Pour plus d'informations, voir la définition de **BIP147** et **OP_CHECKMULTISIG**.*

DUST

Fait référence à des montants de pièces bitcoin extrêmement petits qui sont trop minimes pour être envoyés dans une transaction, car les frais de transaction nécessaires pour les inclure dans un bloc seraient proportionnellement plus élevés que leur valeur. La définition précise de « dust » peut varier selon le contexte, mais il s'agit généralement de toute sortie de transaction qui nécessite plus de frais pour être dépensée qu'elle n'incarne de valeur. Pour l'utilisateur de Bitcoin, il est important de gérer ses UTXO et de pratiquer la consolidation de ceux-ci afin qu'ils ne deviennent pas du Dust.

En français, on pourrait parler de « poussière ».

DUSTING ATTACK

Attaque qui consiste à envoyer de minuscules quantités de bitcoins à un grand nombre d'adresses de réception. L'objectif de l'attaquant est de pousser les destinataires à regrouper ces sommes avec d'autres UTXO. L'attaquant suit ensuite les déplacements futurs de ces faibles quantités de bitcoins, dans le but de former des clusters d'adresses, c'est-à-dire de déterminer si plusieurs adresses appartiennent à une même entité. En croisant les informations recueillies lors d'une dusting attack avec d'autres données et heuristiques utilisées dans l'analyse de chaîne, il est possible pour l'attaquant d'identifier certaines entités et les adresses associées. Cette méthode représente une menace uniquement pour la confidentialité des utilisateurs, mais n'affecte pas la sécurité de leurs fonds.

Certains bitcoiners suggèrent de ne plus utiliser le terme de « dusting attack » car celui-ci induirait en erreur. En effet, le terme de « dust » décrit quelque chose de bien précis dans Bitcoin Core. Si la dusting attack utilisait réellement du dust comme décris dans Core, l'attaque serait inefficace. Certains suggèrent ainsi d'utiliser le terme de « forced address reuse » (réutilisation d'adresse forcée) pour décrire plus précisément cette attaque.

DUST LIMIT

Désigne le seuil en sats en deçà duquel un UTXO est considéré comme de la « poussière » (dust) par un nœud du réseau. Ce seuil fait partie des règles de standardisation qui peuvent être modifiées indépendamment par chaque nœud. Dans Bitcoin Core, cette limite est déterminée par un taux de frais spécifique, fixé par défaut à 3000 sats par kilo-octet virtuel (sats/kvB). Cette limite vise à restreindre la propagation de transactions comprenant de très petits montants en bitcoins. En effet, un UTXO qualifié de poussière implique que son utilisation n'est économiquement pas rationnelle : dépenser cet UTXO coûterait plus cher que de simplement l'abandonner. Si dépenser de la poussière n'est pas rationnel, cela suggère que de telles dépenses ne peuvent être motivées que par des incitations externes, souvent malveillantes. Cela peut notamment poser un problème si un acteur malintentionné cherche à saturer le réseau avec des transactions contenant des montants infimes, dans le but d'accroître la charge opérationnelle des nœuds et potentiellement les empêcher de traiter d'autres transactions légitimes. Pour donner une analogie (un peu bancale, je vous l'accorde), c'est

un peu comme si quelqu'un tentait de payer un panier de courses de 100 € uniquement en pièces de 1 centimes. Pour en savoir plus, je vous recommande de lire les définitions de **DUST**, **DUSTING ATTACK** et de **DUSTRELAYFEE**.

DUSTRELAYFEE

Règle de standardisation utilisée par les nœuds du réseau pour déterminer ce qu'ils considèrent comme la « limite de poussière » (dust limit). Ce paramètre fixe un taux de frais en sats par kilo-octet virtuel (sats/kvB) qui sert de référence pour calculer si la valeur d'un UTXO est inférieure aux frais nécessaires pour l'inclure dans une transaction. En effet, un UTXO est considéré comme « dust » (poussière) sur Bitcoin s'il requiert plus de frais pour être transféré que la valeur qu'il représente lui-même. Le calcul de cette limite est le suivant : `limite de poussière = (taille de l'entrée + taille de la sortie) * taux de frais`. Comme le taux de frais requis pour qu'une transaction soit incluse dans un bloc Bitcoin varie constamment, le paramètre `DustRelayFee` permet de définir le taux de frais utilisé dans ce calcul par chaque nœud. Par défaut, sur Bitcoin Core, cette valeur est fixée à 3000 sats/kvB. Par exemple, pour calculer la limite de poussière d'une entrée et d'une sortie P2PKH, qui mesurent respectivement 148 et 34 octets, le calcul serait : `limite de poussière = (148+34)*3000/1000 = 546 sats`. Cela signifie que le nœud en question ne relayera pas les transactions incluant un UTXO sécurisé en P2PKH dont la valeur est inférieure à 546 sats.

E

ECASH (DAVID CHAUM)

Protocole proposé par David Chaum en 1982, qui est un des premiers systèmes de monnaie numérique conçu pour préserver l'anonymat des utilisateurs. Il repose sur des principes de cryptographie à clé publique pour créer une monnaie numérique qui peut être échangée de manière sécurisée et anonyme. eCash fonctionne par la création de jetons numériques signés par une banque. C'est donc une évolution des banques de dépôt, sans pour autant être décentralisée. Lors des transactions, ces jetons sont transférés entre les parties sans révéler l'identité des utilisateurs, préservant ainsi leur vie privée. eCash est considéré comme un précurseur des cryptomonnaies. Il revient d'ailleurs souvent dans les discussions autour de Bitcoin, certains voulant utiliser des systèmes similaires à eCash en surcouche. Aujourd'hui, la mode est plutôt aux systèmes dits « chaumiens fédérés » comme Fedimint.

ECASH (XEC)

Système de cryptomonnaie, précédemment connu sous le nom de Bitcoin Cash ABC (BCHA), issu d'un hard fork de Bitcoin Cash (BCH). Le fork d'eCash est survenu le 15 novembre 2020 au bloc 661 647, résultant d'un conflit au sein de la communauté Bitcoin Cash.

ECDH

Méthode d'échange de clés cryptographiques basée sur les principes de l'échange de clés Diffie-Hellman, mais qui utilise des courbes elliptiques pour fournir un niveau de sécurité élevé avec des tailles de clés plus petites. Ce protocole permet à deux parties de générer un secret partagé en utilisant leurs paires de clés publiques et privées, sans jamais avoir à échanger les clés privées elles-mêmes. Le secret partagé peut ensuite être utilisé pour sécuriser une communication ultérieure. On retrouve parfois l'utilisation de cet algorithme dans des propositions d'amélioration de Bitcoin, notamment le BIP47 ou le BIP150.

ECDSA (ELLIPTIC CURVE DIGITAL SIGNATURE ALGORITHM)

Algorithme de signature numérique établi sur la cryptographie à courbes elliptiques (ECC). Il s'agit d'une variante de l'algorithme DSA (Digital Signature Algorithm). Il exploite les propriétés des courbes elliptiques pour procurer des niveaux de sécurité comparables à ceux des algorithmes de clé publique traditionnels, tels que RSA, tout en utilisant des clés de taille nettement inférieure. ECDSA permet la génération de paires de clés (clé publique et clé privée) ainsi que la création et la vérification de signatures numériques. Dans le contexte de Bitcoin, ECDSA est utilisé pour dériver des clés publiques, à partir de clés privées. Il est également utilisé pour signer les transactions afin de prouver que l'expéditeur possède les bitcoins. La courbe elliptique utilisée sur Bitcoin au sein d'ECDSA est `secp256k1`, définie par l'équation $y^2 = x^3 + ax + b$. Cet algorithme est celui utilisé dès les débuts de Bitcoin en 2009. Aujourd'hui, il partage sa place avec le protocole de Schnorr, un autre algorithme de signature électronique introduit avec Taproot en 2021.

ECLAIR

Implémentation majeure du protocole Lightning Network écrite en langage Scala. Eclair est développé par la société française Acinq.

> *Attention, « Eclair » était également le nom d'un portefeuille Lightning pour les appareils mobiles, développé par la même société. Aujourd'hui, ce portefeuille n'est plus maintenu.*

ECLIPSE (ATTAQUE)

Attaque qui consiste à isoler et contrôler les communications d'un nœud dans un réseau en créant un environnement artificiel autour de lui. L'objectif est de filtrer ou de manipuler les informations reçues et envoyées par ce nœud, le coupant ainsi de ses pairs légitimes. Dans le cadre de Bitcoin, cette technique peut être utilisée pour induire en erreur un nœud, censurer ou altérer les données qu'il reçoit ou envoie, ou pour mener des attaques de doubles dépenses.

ÉCOLE AUTRICHIENNE

École de pensée économique qui théorise le marché comme un ensemble d'interactions individuelles volontaires, souligne la spontanéité de l'ordre économique et critique les interventions étatiques. L'École Autrichienne défend le rôle de la propriété privée, de la liberté contractuelle, et du libre-échange, tout en critiquant les effets perturbateurs de la création monétaire sur l'économie. Ses contributeurs, tels que Carl Menger, Ludwig von Mises ou Friedrich Hayek, ont travaillé des concepts tels que la formation des prix, la fonction de la monnaie, les dynamiques du capital ou encore la théorie subjective de la valeur. L'École Autrichienne critique le socialisme pour son incapacité à réaliser des calculs économiques efficaces, et favorise une approche libérale. Elle valorise le marché libre et voit dans l'interventionnisme étatique une source de déséquilibres économiques.

ELECTRUM LIGHTNING

Implémentation du Lightning Network écrite en python spécifiquement pour le logiciel Electrum.

ELTOO

Protocole généraliste pour les secondes couches de Bitcoin qui permet de définir la manière de gérer conjointement la propriété d'un UTXO. Eltoo a été conçu par Christian Decker, Rusty Russell et Olaoluwa Osuntokun, notamment pour résoudre les problèmes associés aux mécanismes de négociation de l'état des canaux Lightning, c'est-à-dire entre l'ouverture et la fermeture. L'architecture Eltoo simplifie le processus de négociation en introduisant un système de gestion des états linéaire, remplaçant l'approche basée sur la pénalité par une méthode de mise à jour plus flexible et moins punitive. Ce protocole nécessite l'utilisation d'un nouveau type de SigHash qui permette de ne prendre en compte aucune entrée dans la signature d'une transaction. Ce SigHash a d'abord été appelé `SIGHASH_NOINPUT`, puis `SIGHASH_ANYPREVOUT` (Any Previous Output). Son implémentation est prévue dans le BIP118.

Eltoo est parfois également appelé « LN-Symmetry ».

EMBRANCHEMENT NATUREL

Séparation temporaire de la blockchain résultant de la diffusion quasi simultanée de plusieurs blocs par différents mineurs à une même hauteur. Cette situation se produit lorsque deux blocs, désignés comme A et B, sont trouvés presque simultanément, entraînant une division temporaire du réseau. Puisque chaque nœud considère comme valide le premier bloc qu'il a reçu, mais que tout le monde n'a pas reçu le même bloc en premier, une partie des nœuds suit la chaîne contenant le bloc A, tandis que l'autre suit celle avec le bloc B. Cet embranchement est résolu lorsqu'une des deux chaînes concurrentes dépasse l'autre en termes de travail accumulé. À ce moment, tous les nœuds du réseau s'accordent automatiquement sur la chaîne la plus longue (avec le plus de travail accumulé), un processus que l'on appelle la réorganisation ou la resynchronisation. Ces embranchements naturels sont inhérents au fonctionnement distribué de Bitcoin. Il sont parfaitement normaux et se résolvent spontanément au bout de quelques blocs (généralement un seul).

ENDIANNESS

Désigne l'ordre dans lequel une séquence d'octets est arrangée et interprétée en informatique. On distingue principalement deux types : « big-endian », où l'octet de poids le plus fort (le plus significatif) est stocké en premier, et « little-endian », où l'octet de poids le plus faible (le moins significatif) est stocké en premier.

ENTÊTE DE BLOC

L'entête de bloc est une structure de données servant de composant principal dans la construction d'un bloc Bitcoin. Chaque bloc est composé d'un entête et d'une liste de transactions. L'entête de bloc contient les informations cruciales qui permettent d'assurer l'intégrité et la validité d'un bloc au sein de la chaîne de blocs (blockchain). L'entête de bloc contient 80 octets de métadonnées et se compose des éléments suivants :

- La version du bloc ;
- L'empreinte du bloc précédent ;
- La racine de l'arbre de Merkle des transactions ;
- L'horodatage du bloc ;
- La cible de difficulté ;
- Le nonce (Number only used ONCE).

Par exemple, voici l'entête du bloc n° 785 530 au format hexadécimal, miné par Foundry USA le 15 avril 2023 :

`00e0ff3f5ffe3b0d9247dc437e18edc19252e4517cee941752d5010000000000000000000206bde3a1082`
`6e2acb2f28fba70463601c789293d0c9c4348d7a0d06711e97c0bcb13a64b2e0051743f09a40`

Si l'on décompose cet entête, on peut reconnaitre :

- La version : `00e0ff3f`
- L'empreinte précédente :

`5ffe3b0d9247dc437e18edc19252e4517cee941752d501000000000000000000`

- La racine de Merkle :

`206bde3a10826e2acb2f28fba70463601c789293d0c9c4348d7a0d06711e97c0`

- L'horodatage : `bcb13a64`
- La cible : `b2e00517`
- Le nonce : `43f09a40`

Pour être valide, un bloc doit disposer d'un entête qui, une fois haché avec `SHA256d`, produit un condensat inférieur ou égal à la cible de difficulté.

En anglais, on parle d'un « Block Header ».

ENTRÉE (INPUT)

Dans le contexte de Bitcoin, une « entrée » (ou « input » en anglais) au sein d'une transaction fait référence aux *Unspent Transaction Outputs* (UTXO) utilisés comme fonds d'origine pour une transaction. Chaque entrée contient des références aux UTXO précédents, qui seront alors « consommés

» par la transaction. Ces entrées sont utilisées pour alimenter de nouveaux UTXO qui seront créés comme « sorties » (ou « outputs » en anglais) de la transaction, et qui peuvent ensuite être dépensés dans des transactions futures. Le rôle de la transaction Bitcoin est donc de consommer des UTXO en entrées, et de créer des nouveaux UTXO en sorties. La différence entre les deux correspond aux frais de transactions qui peuvent être récupérés par le mineur gagnant du bloc. D'un point de vue plus large, en informatique, le terme « input » ou « entrée » désigne généralement les données fournies à une fonction, un algorithme, ou un système en tant qu'opérandes ou informations requises pour effectuer une opération ou un calcul. Dans ce sens, le terme est utilisé de manière plus générique pour décrire tout ce qui est fourni à un processus en vue d'obtenir un résultat ou une « sortie » (output). Par exemple, lorsque l'on passe une donnée dans une fonction de hachage cryptographique, cette information est nommée « entrée » ou « input ».

ENTROPIE

L'entropie, dans le contexte de la cryptographie et de l'information, est une mesure quantitative de l'incertitude ou de l'imprévisibilité associée à une source de données ou à un processus aléatoire. L'entropie joue un rôle crucial dans la sécurité des systèmes cryptographiques, notamment dans la génération de clés et de nombres aléatoires. Une entropie élevée garantit que les clés générées sont suffisamment imprévisibles et résistantes aux attaques par force brute, où un attaquant essaie toutes les combinaisons possibles pour deviner la clé. Dans le contexte de Bitcoin, l'entropie est utilisée pour générer des clés privées ou des graines. Lors de la création d'un portefeuille déterministe et hiérarchique, la construction de la phrase mnémonique se fait à partir d'un nombre aléatoire, lui-même issu d'une source d'entropie. La phrase est ensuite utilisée pour générer plusieurs clés privées, de manière déterministe et hiérarchique, afin de créer des conditions de dépense sur des UTXO. Il est essentiel de disposer d'une source d'entropie de qualité pour garantir la sécurité des systèmes cryptographiques. Les sources d'entropie peuvent être des processus physiques, tels que le bruit électronique ou les variations thermiques, ou des processus logiciels, tels que les générateurs de nombres pseudo-aléatoires.

Dans le contexte spécifique de l'analyse de chaîne, l'entropie est également le nom d'un indicateur, dérivé de l'entropie de Shannon, inventé par LaurentMT. Cet indicateur permet de mesurer le manque de connaissance des analystes sur la configuration exacte d'une transaction Bitcoin.

EREBUS (ATTAQUE)

Forme très sophistiquée d'attaque contre le réseau Bitcoin qui permet à un fournisseur de services Internet malveillant d'isoler des nœuds Bitcoin spécifiques. C'est donc une forme d'attaque Eclipse. L'attaque Erebus exploite la structure du réseau Internet, en particulier les points de passage obligés (ou « bottlenecks ») dans le routage entre les systèmes autonomes (AS). Un attaquant, en contrôlant un système autonome, peut manipuler le trafic réseau pour isoler un nœud Bitcoin du reste du réseau, et ainsi lui faire croire à un faux état de la blockchain (blocs ou transactions non connues par le nœud). Cette isolation peut conduire à des doubles dépenses ou de la censure à l'encontre du nœud isolé. Cette attaque est rendue beaucoup plus difficile depuis la version 0.20.0 et l'introduction d'Asmap.

ESMPPS (EQUALIZED SHARED MAXIMUM PAY PER SHARE)

Méthode de calcul de la rémunération des mineurs dans le contexte des pools de minage. ESMPPS vise à répartir la récompense de manière équitable entre toutes les parts, indépendamment du moment de leur soumission ou de la chance de la pool. Cela fonctionne essentiellement comme SMPPS, mais avec cette notion d'égalité pour les parts soumises en plus.

ÉTIQUETAGE

Pratique qui consiste à attribuer une annotation ou une étiquette à un UTXO spécifique dans un portefeuille Bitcoin. Par exemple, si je possède un UTXO provenant d'un achat P2P sur Bisq avec Charles, je pourrais luis attribuer l'étiquette `Non-KYC Bisq Charles`. C'est une bonne pratique qui aide à se rappeler de l'origine ou de la destination prévue de cet UTXO, facilitant ainsi la gestion des fonds et l'optimisation de la confidentialité. L'étiquetage est d'autant plus important lorsqu'il est utilisé avec le coin control. En effet, en permettant aux utilisateurs de différencier et de sélectionner précisément les UTXO pour leurs transactions, cette pratique aide à éviter la fusion d'UTXO provenant de sources différentes. Cela limite les risques associés à l'heuristique d'analyse de chaîne CIOH (*Common Input Ownership Heuristic*), qui peut révéler la propriété commune des entrées d'une transaction.

EXPLORATEUR DE BLOC

Outil en ligne ou en local qui permet de transformer les données brutes de la blockchain Bitcoin en un format structuré et facilement lisible par l'Homme. L'explorateur inclut généralement un moteur de recherche afin de localiser aisément un bloc, une transaction ou une adresse spécifiques.

F

FARADAY

Outil développé par Lightning Labs conçu pour extraire des données d'un nœud LND et les analyser afin d'assister son opérateur. Il offre des recommandations pour la fermeture des canaux non performants et fournit des informations détaillées sur le comportement de routage du nœud. Faraday aide à identifier les canaux à faible volume et ceux ayant des problèmes de disponibilité (uptime). Cet outil vise à assister les opérateurs de nœuds dans l'allocation de leur capital dans leurs canaux.

FEDIMINT

Protocole de paiement et de gestion de bitcoins conçu pour améliorer la confidentialité et réduire les besoins envers la chaîne principale par la mutualisation de la garde des fonds. Fedimint a été créé par Eric Sirion en 2021. Il s'appuie sur un système de banque chaumienne, qui au lieu d'être centralisée sur un seul acteur de confiance, s'appuie sur des fédérations. Ces fédérations sont des groupes de gardiens de confiance qui détiennent collectivement et gèrent les bitcoins des utilisateurs de leur groupe. Au sein du groupe, les utilisateurs peuvent réaliser des paiements avec des billets émis en échange de leur dépôt de bitcoins. L'idée de Fedimint est de déployer ce concept au niveau de communautés locales. C'est donc une sorte d'évolution de la banque de dépôt reposant sur le bitcoin, avec le système eCash de David Chaum, et l'utilisation d'une fédération de personnes de confiances en charge du dépôt et de l'émission du sus-jacent.

FEE SNIPING

Scénario d'attaque dans lequel des mineurs cherchent à réécrire un bloc récemment confirmé dans le but de récupérer les frais de transaction qu'il contient, tout en y ajoutant des transactions à frais élevés arrivées entre-temps dans la mempool. L'objectif final de cette attaque pour le mineur est d'augmenter sa rentabilité. Le fee sniping peut devenir de plus en plus profitable à mesure que la récompense de bloc diminue et que les frais de transaction représentent une part plus importante dans les revenus des mineurs. Elle peut également être avantageuse lorsque les frais contenus dans le bloc précédent sont nettement supérieurs à ceux du meilleur bloc candidat suivant. Pour simplifier, le mineur est face à ce choix en termes d'incitations :

- Miner de manière normale à la suite du dernier bloc, avec une forte probabilité de remporter une récompense faible ;
- Tenter de miner un bloc antérieur (fee sniping), avec une faible probabilité de remporter une récompense élevée.

Cette attaque constitue un risque pour le système Bitcoin, car plus les mineurs l'adoptent, plus les autres mineurs, initialement honnêtes, sont incités à en faire autant. En effet, chaque fois qu'un nouveau mineur s'ajoute à ceux qui tentent un fee sniping, la probabilité qu'une des mineurs attaquants réussisse augmente, et la probabilité qu'un des mineurs honnêtes étende la chaîne diminue en contrepartie. Si cette attaque est menée de manière massive et maintenue dans le temps, les confirmations de bloc ne seraient plus un indicateur fiable de l'immuabilité d'une transaction Bitcoin. Cela rendrait potentiellement le système inutilisable. Pour contrer ce risque, la plupart des logiciels de portefeuille remplissent automatiquement le champ `nLocktime` afin qu'il conditionne la validation de la transaction à l'inclusion dans la prochaine hauteur de bloc. Ainsi, il devient impossible d'inclure la transaction dans une réécriture du bloc précédent. Si l'utilisation massive du `nLocktime` est adoptée par les utilisateurs de Bitcoin, cela réduit considérablement les incitations au fee sniping. En effet, cela encourage la progression de la blockchain plutôt que sa réécriture en réduisant les potentiels bénéfices de celle-ci. Pour les transactions Taproot, le BIP326 propose d'utiliser le champ `nSequence` de manière similaire pour obtenir l'effet équivalent à celui du champ `nLocktime` pour les autres types

de transactions. Cette utilisation permettrait de faire d'une pierre deux coups en améliorant également la confidentialité des protocoles de seconde couche qui utilisent ce champs.

FEE_ESTIMATES.DAT

Fichier dans Bitcoin Core qui stocke des données estimées sur les frais de transaction, compilées par le logiciel à partir des transactions récentes et de l'état actuel de la mempool. Ces statistiques aident l'utilisateur à déterminer des frais appropriés à inclure dans ses transactions pour qu'elles soient confirmées en fonction de ses attentes. Ce fichier existe depuis la version 0.10.

FERME DE MINAGE

Installation où de nombreuses machines de minage (souvent, des ASICs) sont regroupées pour miner du bitcoin en participant au processus de la preuve de travail. Le but de ce regroupement est de faciliter la gestion du parc de machines et de faire des économies d'échelles, notamment pour la mise en place, l'entretien, le refroidissement, la fourniture en électricité et la connexion au réseau.

Attention, la ferme de minage ne doit pas être confondue avec la pool de minage.

FIAT

Monnaie, souvent étatique, dont le cours est imposé par la force publique.

Le terme de « fiat » est parfois traduit par « fiduciaire » bien que ce dernier terme ne prenne pas en compte la dimension de violence légitime qu'incarne le terme « fiat ». En français, il est souvent admis d'utiliser directement le terme anglais de « fiat ».

FIBRE (FAST INTERNET BITCOIN RELAY ENGINE)

Protocole conçu par Matt Corallo en 2016 pour accélérer la propagation des blocs Bitcoin à travers le monde. Son objectif était de réduire les délais de propagation au plus près des limites physiques. FIBRE visait à garantir une distribution plus équitable des opportunités de minage, en s'assurant que la proportion de blocs minés par un participant reflète fidèlement sa contribution en termes de puissance de calcul, peu importe sa situation sur le réseau. En effet, la latence dans la transmission des blocs peut favoriser les grands groupes de mineurs bien connectés au détriment des plus modestes. Ce phénomène pourrait, à terme, augmenter la centralisation du minage et réduire la sécurité globale du système. Pour pallier ce problème, FIBRE introduisait des codes de correction d'erreur et l'envoi de données supplémentaires pour contrebalancer les pertes de paquets, ainsi que l'utilisation de blocs compactés similaires à ceux décrits dans le BIP152, le tout opérant via UDP pour contourner certaines limitations de TCP. Néanmoins, FIBRE fut délaissé en 2020, principalement en raison de sa dépendance à l'égard d'un unique mainteneur et du fait que l'adoption du BIP152 a rendu un tel système moins indispensable.

*Pour plus d'informations, voir la définition de **BIP152**.*

FLAG DAY

Vielle méthode d'activation de soft fork utilisée dans les premières années de Bitcoin. Ce processus défini simplement une date spécifique, connue sous le nom de « Flag Day », à laquelle la mise à jour du protocole doit être adoptée par l'ensemble du réseau. Cette approche est simple et directe : après cette date, les nœuds et les mineurs doivent avoir mis à jour leur logiciel pour se conformer aux nouvelles règles, sans quoi ils risquent de se retrouver sur une chaîne incompatible. Cependant,

cette méthode est très risquée de nos jours, car elle nécessite une coordination et un consensus importants au sein de la communauté, faute de quoi le réseau peut subir une scission, et la chaîne à jour peut ne pas être la longue. La méthode du Flag Day peut toutefois être utilisée pour des changements non controversés ou des rectifications techniques urgentes.

FONCTION DE HACHAGE CRYPTOGRAPHIQUE

Une fonction de hachage, également appelée algorithme de hachage, est une fonction mathématique qui prend une entrée de taille variable (appelée message) et produit une sortie de taille fixe (appelée hash, hachage, condensat ou empreinte). Les fonctions de hachage sont des primitives largement utilisées en cryptographie. Elles présentent des propriétés spécifiques qui les rendent appropriées pour une utilisation dans des contextes sécurisés :

- Résistance aux préimages : Il doit être très difficile de trouver un message donnant un hachage spécifique, c'est-à-dire de trouver une préimage m pour un hash h tel que $h = H(m)$, où H est la fonction de hachage ;
- Résistance aux secondes préimages : Étant donné un message m_1, il doit être très difficile de trouver un autre message m_2 (différent de m_1) tel que $H(m_1) = H(m_2)$;
- Résistance aux collisions : Il doit être très difficile de trouver deux messages distincts m_1 et m_2 tels que $H(m_1) = H(m_2)$;
- Résistance à la falsification : De petites modifications dans l'entrée doivent provoquer des changements significatifs et imprévisibles dans la sortie.

Dans le contexte de Bitcoin, les fonctions de hachage sont utilisées à plusieurs fins, notamment pour le mécanisme de preuve de travail (Proof-of-Work), les identifiants de transaction, la génération d'adresses, le calcul de sommes de contrôle et la création de structures de données telles que les arbres de Merkle. Sur la partie protocolaire, Bitcoin utilise exclusivement la fonction `SHA256d`, également nommée `HASH256`, qui consiste en un double hachage `SHA256`. On utilise aussi `HASH256` dans le calcul de certaines sommes de contrôle, notamment pour les clés étendues (`xpub`, `xprv`…). Sur la partie portefeuille, on utilise également :

- `SHA256` simple pour les sommes de contrôle des phrases mnémoniques ;
- `SHA512` au sein des algorithmes `HMAC` et `PBKDF2` utilisés dans le processus de dérivation des portefeuilles déterministes et hiérarchiques ;
- `HASH160` , qui décrit une utilisation successive d'un `SHA256` et d'un `RIPEMD160` . `HASH160` est utilisé dans le processus de génération des adresses de réception et dans le calcul des empreintes de clés parents pour les clés étendues.

En anglais, on parle de « hash function ».

FONCTIONNAIRE

Dans le cadre de la sidechain Liquid, les fonctionnaires sont des nœuds pilotés par des entités chargées de gérer le système. Ils ont principalement deux rôles : établir le consensus et exécuter des transactions en tant que signataire de bloc (blocksigners) et sécuriser les bitcoins détenus par le réseau afin d'assurer l'ancrage bilatéral (watchmen).

FORCE BRUTE (ATTAQUE)

Méthode de cryptanalyse pour trouver un mot de passe ou une clé qui consiste à essayer par tâtonnement toutes les combinaisons possibles de clés ou de mots de passe jusqu'à trouver celle qui

permet d'accéder à un privilège ou une information protégée. Cette technique repose sur du calcul intensif et peut être extrêmement longue, surtout face à des clés de grande taille. Pour faire face à ce type attaque, il faut utiliser des séquences de mot de passe et de clés plus longues afin de multiplier le nombre d'opérations nécessaires pour l'attaquant. En théorie, la complexité d'une telle attaque est exponentielle en la longueur de la cible.

En anglais, on parle d'une « brute-force attack ».

FORCED ADDRESS REUSE

Certains bitcoiners suggèrent d'utiliser ce terme pour décrire une dusting attack, car ils trouvent que le terme de « dust » est ici inapproprié.

*Pour plus d'informations, voir la définition de **DUST** et **DUST LIMIT**.*

FORK

Le terme de « fork » revêt plusieurs significations dans le cadre de Bitcoin. Il désigne soit une scission du réseau de nœuds en plusieurs groupes séparés, entraînant la création de plusieurs blockchains différentes, soit une modification des règles du protocole, voire les deux simultanément. Pour simplifier, on distingue 4 grandes catégories de forks :

- L'embranchement naturel** : se produit lorsqu'il y a une concurrence temporaire entre deux blocs découverts en même temps à une même hauteur. Cet embranchement peut s'étendre sur plusieurs blocs. Ce type de fork se résout naturellement quand une des chaînes devient plus longue que l'autre (avec plus de travail accumulé), menant à une réorganisation. Cette réorganisation se manifeste avec l'intégralité des nœuds qui s'accordent de nouveau sur une blockchain unique ;
- Le fork de code** : consiste à créer une toute nouvelle cryptomonnaie à partir du code source de Bitcoin, en démarrant une nouvelle blockchain depuis le bloc de genèse ;
- Le hard fork** : représente une modification du protocole Bitcoin, incompatible avec les versions antérieures, en retirant des règles ou en allégeant celles existantes. Cela résulte en la création de deux chaînes distinctes et incompatibles si tous les nœuds ne sont pas mis à jours. Le réseau se scinde alors en deux : ceux qui adoptent les nouvelles règles et ceux qui conservent les anciennes ;
- Le soft fork** : implique des modifications rétrocompatibles qui ajoutent des règles ou rendent plus restrictives celles existantes, sans provoquer de division du réseau. Les nœuds qui n'adoptent pas les nouvelles règles peuvent continuer à suivre la même chaîne que les autres, à condition que la majorité de la puissance de calcul du réseau soutienne la chaîne mise à jour.

  *Pour plus d'informations, voir la définition de **HARD FORK** et **SOFT FORK**.*

FORK (GIT)

Dans le cadre de Git, représente la création d'une copie d'un dépôt existant sur un nouveau compte, permettant ainsi à l'utilisateur de modifier, tester ou développer le projet indépendamment du dépôt original. Les forks permettent la collaboration open source et la contribution à des projets sans affecter le dépôt source.

FORTH

Langage de programmation impératif, conçu pour être simple et efficace, surtout dans les systèmes embarqués et les applications où les ressources sont limitées. Forth se distingue par son style de programmation. Il utilise une pile pour le stockage des données et des calculs. Le langage Script utilisé sur Bitcoin a un fonctionnement similaire à Forth.

FPPS (FULL PAY PER SHARE)

Méthode de calcul de la rémunération des mineurs dans le contexte des pools de minage. C'est une évolution de la méthode Pay Per Share (PPS). Elle rémunère les mineurs non seulement pour chaque part valide qu'ils soumettent, mais inclut également une part des frais de transaction du réseau. La rémunération est calculée sur la base des transactions moyennes précédentes et du hashrate de la pool. Ainsi, les mineurs reçoivent une rétribution pour les parts soumises, qu'un bloc soit trouvé ou non. Cette méthode rémunère la valeur attendue. Elle offre une rémunération stable et prévisible pour les mineurs, car elle élimine la variabilité liée à la probabilité de trouver un bloc. Toutefois, elle est plus risquée pour les opérateurs de pool, car ils doivent payer les mineurs même lorsqu'aucun bloc n'est trouvé, absorbant ainsi le risque de variance.

FRAIS DE TRANSACTION

Les frais de transaction représentent une somme qui vise à rémunérer les mineurs pour leur participation au mécanisme de la preuve de travail. Ces frais incitent les mineurs à inclure les transactions dans les blocs qu'ils créent. Ils sont le résultat de la différence entre le montant total des inputs et le montant total des outputs d'une transaction. $frais = inputs - outputs$

Ils sont exprimés en `sats/vBytes`, ce qui veut dire que les frais ne dépendent pas du montant des bitcoins envoyés, mais du poids de la transaction. Ils sont choisis librement par l'émetteur d'une transaction et déterminent la vitesse d'inclusion de la transaction dans un bloc par un mécanisme d'enchère. Par exemple, imaginons que je réalise une transaction avec un input de `100 000 sats`, un output de `40 000 sats` et un output de `58 500 sats`. Le total des outputs est de `98 500 sats`. Les frais alloués à cette transaction sont de `1 500 sats`. Le mineur qui inclut ma transaction pourra créer `1 500 sats` dans sa transaction coinbase en contrepartie des `1 500 sats` que je n'ai pas récupérés dans mes outputs.

Les transactions avec des frais plus élevés, en fonction de leur taille, sont traitées en priorité par les mineurs, ce qui peut accélérer le processus de confirmation. Inversement, les transactions avec des frais plus faibles peuvent être retardées lors des périodes de forte congestion. Il convient de noter que les frais de transaction Bitcoin sont distincts de la subvention de bloc, qui est une incitation supplémentaire pour les mineurs. La récompense de bloc est composée de nouveaux bitcoins créés à chaque bloc miné (subvention de bloc), ainsi que des frais de transaction collectés. Tandis que la subvention de bloc diminue au fil du temps en raison de la limitation de l'offre totale de bitcoins, les frais de transaction, eux, continueront de jouer un rôle crucial pour encourager les mineurs à participer.

Au niveau protocolaire, rien n'empêche les utilisateurs d'inclure des transactions sans aucuns frais dans un bloc. En réalité, ce type de transaction sans frais fait exception. Par défaut, les nœuds Bitcoin ne relaient pas les transactions disposant de frais inférieurs à `1 sat/vBytes`. Si certaines transactions sans frais ont pu passer, c'est parce qu'elles ont été intégrées directement par le mineur gagnant, sans parcourir le réseau de nœuds. Par exemple, la transaction `fd456524104a6674693c29946543f8a0befccce5a352bda55ec8559fc630f5f3` n'inclut aucuns frais. Dans cet exemple précis, c'était une transaction initiée par le directeur de la pool de minage F2Pool. En tant qu'utilisateur normal, la limite inférieure est donc actuellement de `1 sat/vBytes`. Il convient

également de tenir compte les limites de purge. En période de forte congestion, les mempools des nœuds purgent leurs transactions en attente en dessous d'un certain seuil, afin de respecter leur limite de RAM attribuée. Cette limite est librement choisie par l'utilisateur, mais beaucoup laissent la valeur Bitcoin Core par défaut. Pour le moment, cette limite est de 300 Go par défaut, elle peut être modifiée dans le fichier `bitcoin.conf` avec le paramètre `maxmempool`.

En anglais, on parle de « transaction fees ».

G

GAP LIMIT

Paramètre utilisé dans les logiciels de portefeuille Bitcoin pour déterminer le nombre maximal d'adresses consécutives non utilisées à générer avant de cesser la recherche de transactions supplémentaires. L'ajustement de ce paramètre est souvent nécessaire lors de la récupération d'un portefeuille pour garantir que toutes les transactions soient bien trouvées. Un Gap Limit insuffisant pourrait entraîner l'omission de certaines transactions si des adresses étaient ignorées lors des phases de dérivation. Augmenter le Gap Limit permet au portefeuille de rechercher plus loin dans la séquence d'adresses, afin de récupérer toutes les transactions associées. En effet, une seule `xpub` peut théoriquement dériver plus de 4 milliards d'adresses de réception (adresses internes et externes). Toutefois, les logiciels de portefeuille ne peuvent pas toutes les dériver et vérifier leur usage sans engendrer un coût en ressources énorme. Ainsi, ils procèdent par ordre d'index, car c'est normalement dans cet ordre que tous les logiciels de portefeuille vous les génèrent. Le logiciel enregistre chaque adresse utilisée avant de passer à la suivante, et il cesse sa recherche lorsqu'il rencontre un nombre d'adresses consécutivement vides. Ce nombre, c'est ce que l'on appelle le Gap Limit. Si par exemple, le Gap Limit est fixé à `20`, et que l'adresse `m/84'/0'/0'/0/15/` est vide, le portefeuille dérivera les adresses jusqu'à `m/84'/0'/0'/0/34/`. Si cette plage d'adresses reste inutilisée, la recherche s'arrête là. Par conséquent, une transaction utilisant l'adresse `m/84'/0'/0'/0/40/` ne serait pas détectée dans cet exemple.

GENÈSE (BLOC)

Le bloc de genèse Bitcoin, également connu sous le nom de bloc Genesis ou bloc #0, est le premier bloc du système Bitcoin. Il incarne le lancement concret de Bitcoin. Le bloc de genèse a été créé par le fondateur anonyme de Bitcoin, Satoshi Nakamoto, le 3 janvier 2009. Son hash est `000000000019d6689c085ae165831e934ff763ae46a2a6c172b3f1b60a8ce26f`. Ce bloc contient seulement une transaction coinbase qui génère 50 bitcoins en récompense pour le mineur (dans ce cas, Satoshi Nakamoto lui-même). Il est particulièrement significatif en raison de son message incorporé dans la transaction coinbase : `The Times 03/Jan/2009 Chancellor on brink of second bailout for banks`. Cette citation est une référence à un article du journal *The Times*. Le message est interprété comme une critique du système financier traditionnel et de ses dérives, ce qui a en partie motivé la création de Bitcoin en tant qu'alternative. Puisqu'il incarne le tout premier bloc de la blockchain Bitcoin, le bloc de genèse ne possède évidemment pas de champ contenant le hachage du bloc antérieur (car il n'y en pas). Par ailleurs, les 50 bitcoins générés en récompense dans ce bloc ne sont pas dépensables au niveau protocolaire.

> *J'aime différencier les bitcoins perdus pour cause protocolaire, des bitcoins perdus pour cause applicative. Par définition, les bitcoins perdus au niveau protocolaire ne seront jamais dépensables, sauf à refaire la preuve de travail postérieure. J'y inclus notamment les pertes liées à la non-réclamation de la récompense coinbase, ou celles liées à un script OP_RETURN. Au contraire, les bitcoins perdus au niveau applicatif, souvent pour cause de perte de clés, seront sûrement un jour débloqués à cause des limitations de la cryptographie employée.*

GETWORK

Ancien protocole de minage pour Bitcoin créé en 2010 par m0mchil. Getwork permettait aux mineurs de recevoir des données de travail de la part d'un nœud complet. Il était établi sur des requêtes RPC permettant d'obtenir des en-têtes de blocs à travailler pour trouver une preuve de travail valide. Getwork était optimisé pour le minage par GPU. Ce fut le premier logiciel open source conçu pour optimiser la communication entre les nœuds et les mineurs à une époque ou quelques acteurs gardait

ces logiciels privés. Getwork a été progressivement remplacé par Stratum, plus efficace, notamment pour les ASIC, et moins gourmand en bande passante.

GIT

Système de contrôle de version distribué conçu pour gérer tout type de projet logiciel avec efficacité. Il permet aux développeurs de suivre les modifications apportées au code source d'un projet au fil du temps, de revenir à des états antérieurs, de gérer des branches et de fusionner des modifications. Git facilite la collaboration entre les développeurs en permettant à plusieurs personnes de travailler sur le même projet simultanément, sans risque de conflit dans les fichiers. Chaque développeur travaille localement et peut ensuite synchroniser ses modifications avec le dépôt central. Créé en 2005 par Linus Torvalds, Git est devenu de fait le standard pour le contrôle de version dans l'industrie du logiciel. Les développements des implémentations de nœud Bitcoin, dont Bitcoin Core, sont gérées avec Git.

GO (GOLANG)

Langage de programmation développé par Google, connu pour sa simplicité et son efficacité. Go est particulièrement adapté pour les applications cloud, les services web, et les systèmes distribués, grâce à sa gestion native de la concurrence et son modèle de programmation facile à comprendre.

GOLDFINGER (ATTAQUE)

Voir la définition de **ATTAQUE DES 51%**.

GOSSIP

Dans le cadre de Lightning, c'est un protocole de communication entre les nœuds pour partager les informations sur l'état actuel et la topologie du réseau. Le protocole de gossip permet aux nœuds de connaître l'état des canaux de paiement et des autres nœuds, facilitant le routage des transactions à travers le réseau sans nécessiter de connexions directes entre tous les nœuds. Le gossip assure une diffusion fiable et cohérente des données à tous le monde, malgré la nature dynamique du réseau.

En français, on pourrait traduire « gossip protocol » par « protocole de bavardage ».

GRAINE (SEED)

Dans le cadre spécifique d'un portefeuille déterministe hiérarchique Bitcoin, une graine (ou « seed » en anglais) est une information de 512 bits issue d'un aléa. Elle permet de générer de manière déterministe et hiérarchique un ensemble de clés privées, et leurs clés publiques correspondantes, pour un portefeuille Bitcoin. La graine (seed) est souvent confondue avec la phrase de récupération en elle-même. Pourtant, c'est une information différente. La graine est obtenue en appliquant la fonction `PBKDF2` sur la phrase mnémonique et sur l'éventuelle passphrase. La graine a été inventée avec le BIP32 qui définit les bases du portefeuille déterministe hiérarchique. Dans ce standard, la graine mesurait 128 bits. Cela permet de dériver toutes les clés d'un portefeuille depuis une information unique, contrairement aux portefeuilles JBOK (Just a Bunch Of Keys) qui nécessitent de réaliser de nouvelles sauvegardes pour toute clé générée. Le BIP39 est par la suite venu proposer un encodage de cette graine, afin de simplifier sa lecture par l'humain. Cet encodage se fait sous la forme d'une phrase, que l'on nomme généralement phrase mnémonique ou phrase de récupération. Ce standard permet d'éviter les erreurs au niveau de la sauvegarde de la graine, notamment grâce à l'utilisation d'une somme de contrôle.

De manière plus générale, en cryptographie, une graine est un morceau de données aléatoires utilisé comme point de départ pour générer des clés cryptographiques, des chiffrements ou des séquences aléatoires. La qualité et la sécurité de nombreux processus cryptographiques dépendent de la nature aléatoire et de la confidentialité de la graine.

> *La traduction anglaise de « graine » est « seed ». En français, beaucoup utilisent directement le mot anglais pour désigner la graine.*

GUI

Acronyme de « Graphical user interface », ou « interface graphique utilisateur » en français. C'est une forme d'interface utilisateur qui permet d'interagir avec des logiciels à travers des éléments visuels interactifs (boutons, icônes, images, fenêtres…) et qui privilégie l'utilisation de dispositifs de pointage (la souris) plutôt que de commandes textuelles comme avec la CLI.

GUISETTINGS.INI.BAK

Fichier dans Bitcoin Core utilisé pour stocker une sauvegarde des paramètres de l'interface graphique (GUI). Cette sauvegarde est créée lors de l'utilisation de l'option `-resetguisettings`, qui réinitialise les paramètres de la GUI à leurs valeurs par défaut. Ce fichier permet à l'utilisateur de restaurer ses configurations précédentes si nécessaire.

H

HAL FINNEY

Cryptographe et développeur informatique de renom, Hal Finney est célèbre pour son rôle crucial dans les débuts de Bitcoin et ses contributions à la cryptographie. Dès la publication du White Paper de Bitcoin en 2008, il fut l'un des premiers à interagir avec Satoshi Nakamoto. Il apporte des retours, signale des bugs et propose des améliorations après le lancement du logiciel en janvier 2009. Il a marqué l'histoire de Bitcoin en étant le destinataire de la première transaction Bitcoin, recevant 10 BTC de Satoshi dans le bloc n°170. Hal Finney est aussi probablement la première personne, après Satoshi, à avoir miné un bloc : le bloc n°78. Plus que cela, Hal Finney a été le premier promoteur de Bitcoin durant une période où le projet était encore méconnu. En dehors de Bitcoin, il est reconnu pour son invention de RPoW (*Reusable Proofs of Work*), un système de monnaie électronique lancé en 2004. Bien que RPoW n'ait pas rencontré le succès attendu, il demeure l'un des précurseurs les plus aboutis de Bitcoin. En tant que cypherpunk engagé, Hal Finney a également joué un rôle déterminant dans l'élaboration et l'amélioration de PGP (*Pretty Good Privacy*). Hal Finney nous a quittés le 28 août 2014, emporté par la sclérose latérale amyotrophique (maladie de Charcot). Il a été cryogénisé par la fondation Alcor. Il restera une figure majeure de l'histoire de la cryptographie et de Bitcoin.

HALVING

Le terme « halving » (division par deux) fait référence à un événement programmé qui réduit de moitié la récompense attribuée aux mineurs pour chaque bloc miné via la subvention de bloc. Cette réduction s'applique spécifiquement à la partie de la subvention de bloc constituée de nouveaux bitcoins créés ex-nihilo. Le halving a été conçu par Satoshi Nakamoto, le créateur de Bitcoin, comme un mécanisme permettant de contrôler l'inflation et d'assurer un approvisionnement limité en bitcoins. La récompense de bloc initiale était de 50 bitcoins, et le halving se produit tous les 210 000 blocs minés, ce qui prend environ quatre ans. Le premier halving a eu lieu en novembre 2012, réduisant la récompense de bloc à 25 bitcoins, et les suivants ont réduit la récompense à 12,5, puis 6,25 bitcoins respectivement. Les halvings continueront à se produire jusqu'à ce que la récompense de bloc atteigne zéro, moment auquel l'offre maximale de 21 millions de bitcoins aura été (presque) atteinte. Le prochain halving de Bitcoin devrait avoir lieu aux alentours du printemps 2024, bien que la date exacte puisse légèrement varier en fonction du temps de minage des blocs. À ce moment-là, la récompense de bloc sera réduite de 6,25 à 3,125 bitcoins. Ce sera le quatrième halving de l'histoire de Bitcoin. Les bitcoiners étudient attentivement les effets des halvings sur le système, car les incitations pour les mineurs diminuent avec le temps. À mesure que les récompenses de bloc baissent, les frais de transaction deviennent une source de revenus de plus en plus importante pour les mineurs, ce qui garantit leur motivation à continuer à participer à la preuve de travail.

HARD FORK

Modification des règles du protocole de manière non rétrocompatible. Cette modification donne lieu à une séparation définitive du réseau de nœuds Bitcoin en deux groupes distincts : les nœuds avec la mise à jour et les nœuds sans la mise à jour. Cette scission se matérialise par la division de la blockchain originale en deux blockchains distinctes, partageant toutefois un historique commun, d'où l'usage du terme « fork », traduisible en français par « fourchette ». Une modification est dite non rétrocompatible lorsqu'elle supprime ou rend moins restrictives certaines règles du protocole. En d'autre terme, un hard fork s'observe lorsque certains nœuds font en sorte qu'un bloc invalide devienne valide. En résulte alors la formation d'une nouvelle version du protocole, qui peut soit remplacer le Bitcoin original si une majorité est trouvée, soit devenir un altcoin indépendant s'il n'est qu'utilisé en marge. Par exemple, Bitcoin Cash (BCH) est un hard fork de Bitcoin. L'embranchement a eu lieu au bloc n° 478 559, le 1er août 2017.

HARDWARE WALLET

Un hardware wallet, ou portefeuille matériel, est un dispositif électronique dédié à la sécurisation et à la gestion des clés privées d'un portefeuille Bitcoin. Ces périphériques sont conçus pour procurer une sécurité renforcée par rapport aux portefeuilles logiciels qui résident sur des machines polyvalentes et directement connectées à internet. Les hardwares wallets stockent la phrase mnémonique hors ligne, sur un matériel qui dispose d'une infime surface d'attaque, ce qui l'isole des environnements potentiellement vulnérables. Lorsqu'une transaction est effectuée, le portefeuille matériel signe la transaction à l'intérieur du dispositif lui-même, sans exposer la clé privée à l'extérieur. Une fois la transaction signée, elle est transmise au réseau Bitcoin pour être confirmée et incluse dans la blockchain Bitcoin. Parmi les modèles de hardwares wallets les plus populaires, on peut citer : Ledger, Trezor, Coldcard, Passport, BitBox, Satochip, Jade ou encore SeedSigner (liste non exhaustive).

> *Le hardware wallet peut être exprimé de différentes manières en français. Certains parlent de « portefeuille matériel » ou bien de « portefeuille froid ». Certains bitcoiners préfèrent que l'on emploie le terme de « périphérique de signature », ou « signing device » en anglais, afin d'éviter de faire penser que les bitcoins se trouvent physiquement dans le portefeuille.*

HARDWARE WALLET INTERFACE (HWI)

Interface standardisée permettant l'intégration et l'interaction entre des logiciels de gestion de portefeuilles Bitcoin et des portefeuilles matériels (hardware wallets). Plus précisément, HWI est à la fois une bibliothèque en Python et un outil en ligne de commande. Il facilite la communication entre ces composants en utilisant des PSBTs (transactions Bitcoin partiellement signées) et éventuellement des Descriptors (output script descriptors). Cette interface a d'abord été développée pour Bitcoin Core, puis, elle est devenue un standard utilisé par la plupart des logiciels de portefeuilles.

HASHCASH

Système de preuve de travail conçu par Adam Back en 1997, principalement pour lutter contre le spam et les attaques DoS. Il repose sur le principe qu'un expéditeur doit effectuer un travail de calcul (spécifiquement, la recherche d'une collision partielle sur une fonction de hachage cryptographique) pour prouver son travail. Cette tâche est coûteuse en temps et en énergie pour l'expéditeur, mais la vérification du résultat par le destinataire est rapide et simple. Ce protocole s'est révélé particulièrement adapté à la lutte contre le spam dans les messageries électroniques, car il est peu contraignant pour les utilisateurs légitimes, tout en constituant un obstacle majeur pour les spammeurs. En effet, l'envoi d'un seul courriel requiert quelques secondes de calcul, mais reproduire cette opération des millions de fois rend l'opération extrêmement coûteuse en termes d'énergie et de temps, ce qui vient souvent annuler l'intérêt économique des campagnes de spam, qu'elles soient à but marketing ou malveillant. De plus, il permet de préserver l'anonymat de l'expéditeur. HashCash a rapidement été adopté par des cypherpunks qui cherchaient à développer un système de monnaie électronique anonyme sans intermédiaire. En effet, l'innovation d'Adam Back a introduit pour la première fois la notion de rareté dans le monde numérique. On retrouve alors le concept de preuve de travail dans plusieurs propositions de monnaie électronique antérieures à Bitcoin, dont :

- b-money de Wei Dai publié en 1998 ;
- R-POW de Hal Finney publié en 2004 ;
- BitGold de Nick Szabo publié en 2005.

Le principe de HashCash se retrouve également au sein du protocole, où il est utilisé comme mécanisme de protection face aux attaques Sybil.

HASHRATE

Indicateur de la puissance de calcul du réseau, mesurée en hachages par seconde (H/s). Il indique la capacité des mineurs à exécuter des opérations de hachage dans le cadre de la preuve de travail. Un hashrate élevé signifie une plus grande sécurité de l'historique économique de Bitcoin et une plus grande résistance aux attaques, car il faudrait une quantité substantielle de puissance de calcul pour compromettre le réseau. Le hashrate est également indicatif de la concurrence entre les mineurs : plus le hashrate est élevé, plus la difficulté de minage est grande, ce qui influence la récompense et donc la rentabilité des mineurs. C'est donc un indicateur clé de la santé et de la sécurité du système Bitcoin. De la même manière que le hashrate sert à mesure le taux de hachage global du réseau Bitcoin, il peut également être utilisé pour mesurer le taux de hachage d'une machine, d'une ferme de minage ou encore d'une pool de minage.

> *En français, on parle de « taux de hachage », bien que le terme de « hashrate » soit largement utilisé dans le langage courant.*

HMAC-SHA512

`HMAC-SHA512` est l'acronyme de « Hash-based Message Authentication Code - Secure Hash Algorithm 512 ». C'est un algorithme cryptographique utilisé pour vérifier l'intégrité et l'authenticité des messages échangés entre deux parties. Il combine la fonction de hachage cryptographique `SHA512` (Secure Hash Algorithm 512) avec une clé secrète partagée pour générer un code d'authentification de message (MAC) unique pour chaque message. Dans le contexte de Bitcoin, l'utilisation naturelle de `HMAC-SHA512` est légèrement dérivée. On utilise cet algorithme dans le processus de dérivation déterministe et hiérarchique de l'arbre de clés cryptographiques d'un portefeuille. `HMAC-SHA512` est notamment utilisé pour passer de la graine (seed) à la clé maîtresse, puis pour chaque dérivation d'une paire parent vers des paires enfants. On retrouve également cet algorithme au sein d'un autre algorithme de dérivation, nommé `PBKDF2`, utilisé pour passer de la phrase de récupération et de la passphrase à la graine.

HORODATAGE (TIMESTAMP)

L'horodatage, ou « timestamp » en anglais, est un mécanisme qui consiste à associer un repère temporel précis à un événement, une donnée ou un message. Dans le contexte généraliste des systèmes informatiques, l'horodatage sert à déterminer l'ordre chronologique des opérations et à vérifier l'intégrité des données en fonction du temps. Dans le cas spécifique de Bitcoin, les horodatages permettent d'établir la chronologie des transactions et des blocs. Chaque bloc dans la blockchain contient un horodatage indiquant le moment approximatif de sa création. Satoshi Nakamoto parle même d'un « serveur d'horodatage », dans son White Paper, pour décrire ce que l'on appellerait aujourd'hui la « blockchain ». Le rôle de l'horodatage sur Bitcoin est de déterminer la chronologie des transactions, afin de pouvoir déterminer, sans l'intervention d'une autorité centrale, quelle transaction est arrivée en première. Ce mécanisme permet à chaque utilisateur de vérifier la non-existence d'une transaction par le passé, et donc d'éviter qu'un utilisateur malintentionné opère une double dépense. Ce mécanisme est justifié par Satoshi Nakamoto dans son White Paper par la célèbre phrase : « *Le seul moyen pour confirmer l'absence d'une transaction est d'être au courant de toutes les transactions.* » Cette norme est établie sur l'heure Unix, qui représente le total de secondes passées depuis le premier janvier 1970.

> *L'horodatage des blocs est relativement flexible sur Bitcoin, car pour qu'un horodatage soit considéré comme valide, il est simplement nécessaire qu'il soit plus grand que le temps médian des 11 blocs qui le précèdent (MTP) et plus petit que la médiane des temps retournés par les nœuds (network-adjusted time) plus 2 heures.*

I

INBOUND CAPACITY

Désigne la quantité maximale de bitcoins qu'un nœud peut recevoir à travers un canal spécifique sur le Lightning Network. Elle dépend des fonds que le nœud pair a engagés dans le canal lors de son ouverture, ou que l'on a envoyé lors d'un paiement Lightning sortant.

En français, on peut le traduire par « capacité entrante ».

INDEXES/TXINDEX/

Fichiers dans Bitcoin Core qui sont dédiés à l'indexation de toutes les transactions présentes dans la blockchain. Cette indexation permet de rechercher rapidement des informations détaillées sur n'importe quelle transaction en utilisant son identifiant (TXID), sans avoir à parcourir l'intégralité de la blockchain. La création de ces fichiers d'indexation est une option non activée par défaut dans Bitcoin Core. Si cette fonctionnalité n'est pas activée, votre nœud indexera uniquement les transactions associées aux portefeuilles rattachés à votre nœud. Pour activer l'indexation de toutes les transactions, il faut régler le paramètre `-txindex=1` dans le fichier bitcoin.conf. Cette option est particulièrement utile pour les applications et services qui font des recherches fréquentes dans l'historique des transactions de Bitcoin.

INITIAL BLOCK DOWNLOAD (IBD)

Fait référence au processus par lequel un nœud télécharge et vérifie la blockchain depuis le bloc Genesis, et se synchronise aux autres nœuds du réseau Bitcoin. L'IBD doit être réalisée au lancement d'un nouveau nœud complet Bitcoin. Lors du lancement de cette synchronisation initiale, le nœud ne dispose d'aucune information sur les transactions précédentes. Il télécharge donc chaque bloc depuis les autres nœuds du réseau, vérifie sa validité, puis l'ajoute à sa version locale de la blockchain. Il convient de noter que l'IBD peut être longue et exigeante en ressources en raison de la taille croissante de la blockchain et de l'UTXO set. La rapidité de son exécution dépend des capacités de calcul de l'ordinateur qui héberge le nœud, de ses capacités en RAM, de la vitesse du dispositif de stockage et de la bande passante. Pour vous donner une idée, si vous disposez d'une connexion internet puissante, et que le nœud est hébergé sur le dernier MacBook avec beaucoup de RAM, l'IBD ne prendra que quelques heures. En revanche, si vous utilisez un micro-ordinateur comme un Raspberry Pi, l'IBD pourra prendre une semaine ou plus.

En français, il est globalement admis de parler directement d'un(e) IBD. La traduction parfois employée est « synchronisation initiale », ou « téléchargement initial des blocs ».

INDEX (NUMÉRO DE CLÉ)

Dans le contexte d'un portefeuille HD (Hierarchical Deterministic), fait référence au numéro séquentiel attribué à une clé enfant générée à partir d'une clé parent. Cet index est utilisé en combinaison avec la clé parent et le code chaîne parent pour dériver de manière déterministe des clés enfants uniques. Il permet une organisation structurée et la génération reproductible de multiples paires de clés enfants sœurs depuis une même clé parent. C'est un entier de 32 bits utilisé dans la fonction de dérivation `HMAC-SHA512`. Ce nombre permet donc de différencier les clés enfants sœurs au sein d'un portefeuille HD.

INPUT

Fait référence aux *Unspent Transaction Outputs* (UTXO) utilisés comme fonds d'origine pour une transaction.

*Pour plus d'informations voir la définition d'**ENTRÉE**.*

IP_ASN.MAP

Fichier utilisé dans Bitcoin Core pour stocker l'ASMAP qui permet d'améliorer le bucketing (c'est-à-dire, le regroupement) des adresses IP, en se basant sur les numéros de systèmes autonomes (ASN). Plutôt que de regrouper les connexions sortantes par préfixes de réseau IP (/16), ce fichier permet de diversifier les connexions en établissant une carte d'adressage IP vers les ASN, qui sont des identifiants uniques pour chaque réseau sur Internet. L'idée est d'améliorer la sécurité et la topologie du réseau Bitcoin en diversifiant les connexions pour se prémunir contre certaines attaques (notamment l'attaque Erebus).

ISSUE

Dans le cadre de Github et d'autres plateformes d'hébergement de code, une issue est un rapport qui signale un bug, propose une amélioration ou suggère une nouvelle fonctionnalité. Elle sert de point de discussion pour les contributeurs et permet de suivre les tâches à accomplir ou les problèmes à résoudre dans le projet. En tant qu'utilisateur de Bitcoin, vous pouvez aider les logiciels open source que vous utilisez en signalant des éventuels bugs via des issues sur le dépôt du projet.

J

JAVA

Langage de programmation polyvalent orienté objet, célèbre pour sa philosophie « écrire une fois, exécuter partout ». Java est largement utilisé pour le développement d'applications d'entreprise, de logiciels mobiles (en particulier pour Android), et dans des systèmes embarqués ou d'applications serveur.

JAVASCRIPT (NODE.JS)

JavaScript est un langage de programmation principalement utilisé pour le développement web. Node.js est un environnement d'exécution JavaScript côté serveur, permettant de créer des applications web évolutives. Ensemble, ils offrent une solution complète pour la programmation d'applications web.

JBOK (PORTEFEUILLE)

Les portefeuilles JBOK, acronyme pour « Just a Bunch Of Keys » (en français « juste un trousseau de clés »), font référence aux portefeuilles Bitcoin initiaux qui stockaient un ensemble de paires de clés générées de manière indépendante et pseudo-aléatoire. Contrairement aux portefeuilles HD (Hierarchical Deterministic) modernes, qui génèrent des clés de manière déterministe et hiérarchique à partir d'une graine unique, les portefeuilles JBOK ne présentaient aucune relation hiérarchique ou déterministe entre les clés. Elles étaient toutes indépendantes les unes des autres. En raison de leur gestion moins efficace et de la difficulté de sauvegarde, ces portefeuilles sont devenus obsolètes et ont été spontanément remplacés par des solutions HD plus avancées, comme standardisées dans le BIP32.

K

KNAPSACK SOLVER

Ancienne méthode utilisée pour la sélection de pièces dans le portefeuille de Bitcoin Core avant la version 0.17. Le Knapsack Solver tente de résoudre le problème de sélection de pièces en sélectionnant de manière itérative et aléatoire des UTXO et en les additionnant par sous-ensembles, dans l'objectif de minimiser les frais et la taille de la transaction.

KYC (KNOW YOUR CUSTOMER)

Procédure réglementaire utilisée par certaines entreprises opérant sur Bitcoin pour vérifier l'identité de leurs clients dans le cadre de la lutte contre le blanchiment d'argent et le financement du terrorisme. Le KYC implique la collecte et la vérification de données personnelles. Dans le cadre de l'achat de bitcoins, le KYC amène plusieurs risques pour l'utilisateur, notamment :

- Le risque de fuite de données personnelles en lien avec une activité sur Bitcoin : Le stockage d'informations sur les serveurs d'entreprises peut entraîner des fuites, exposant les données des utilisateurs à des tentatives d'hameçonnage, des attaques physiques, ou une usurpation d'identité, notamment en raison de leur association avec l'environnement de Bitcoin ;
- L'exposition à la surveillance étatique : L'achat de BTC via des acteurs régulés peut révéler à l'État que l'utilisateur a possédé du bitcoin à un moment donné, ce qui pourrait avoir des répercutions futures en cas de bouleversement politiques ou économiques ;
- La facilitation du traçage on-chain : La réalisation d'un KYC crée un lien direct entre l'identité de l'utilisateur et ses transactions sur la blockchain, permettant d'établir un point d'entrée pour une analyse de chaîne.

L

LABEL

Étiquette ou annotation attribuée à un UTXO afin de se souvenir de sa provenance.

*Pour plus d'informations, voir la définition de **ÉTIQUETAGE**.*

LCB/FT

La lutte contre le blanchiment des capitaux et le financement du terrorisme (LCB/FT) fait référence aux mesures réglementaires adoptées pour prévenir l'utilisation de Bitcoin dans des activités illégales. Ces mesures incluent l'identification et la vérification de l'identité des clients (KYC), la surveillance des transactions pour détecter des schémas « suspects », et la collaboration avec les autorités pour signaler des activités considérées comme illégales. Les plateformes d'échange régulées sont tenues de s'y conformer pour opérer dans de nombreuses juridictions, notamment en France.

LDK (LIGHTNING DEV KIT)

Kit de développement (SDK) pour Lightning. LDK est une collection de bibliothèques et d'outils destinés aux développeurs pour intégrer facilement Lightning à leurs logiciels ou pour créer des applications Lightning en réduisant la complexité. LDK gère les aspects complexes de l'intégration de fonctionnalités liées à Lightning. Ce projet a été lancé par Spiral, une entreprise créée par Jack Dorsey, et s'est basée sur Rust-Lightning (RL).

LEVELDB

Bibliothèque de stockage de clés-valeurs légère, rapide et open-source, conçue par Google. On l'utilise sur Bitcoin pour stocker l'UTXO set, l'index des transactions et l'index des blocs. Ce système de base de données a été introduit en 2012 dans le cadre de la Pull Request « *Ultraprune* » visant à remplacer BerkeleyDB. Ce changement a eu des répercussions significatives, notamment la création d'une première division de la blockchain avec une réorganisation majeure de 24 blocs le 12 mars 2013. Cet incident a été détaillé dans le BIP50. Plus tard, ce changement de système a même conduit à un hard fork non intentionnel le 15 mai 2013.

LIGHTNING LABS

Entreprise spécialisée dans le développement sur le Lightning Network. Fondée en 2016, elle est à l'origine de Lightning Network Daemon (LND), une des implémentations majeures du protocole. Lightning Labs est également à l'origine des services Pool, Loop et Faraday. Plus récemment, ils ont annoncé travailler sur le protocole Taproot Assets (anciennement TARO).

LIGHTNING NETWORK

Protocole de couche supérieure, construit au-dessus du protocole Bitcoin, visant à permettre des transactions rapides et à faible coût. Il permet la création de canaux de paiement entre les participants, au sein desquels les transactions peuvent être effectuées presque instantanément et avec des frais minimes, sans avoir à enregistrer chaque transaction individuellement sur la blockchain. Les canaux peuvent rester ouverts quasi indéfiniment, et ne nécessitent des transactions sur la blockchain que lors de leur ouverture et de leur clôture. Le Lightning Network vise à améliorer la scalabilité de Bitcoin et à rendre possible son utilisation pour des paiements de faible valeur. Toutefois, le Lightning Network n'est pas une solution parfaite. Ce protocole a une tendance naturelle à la centralisation sur de gros nœuds institutionnels. Il peut également être difficile de l'utiliser durant les périodes de très

fortes congestions, comme on a pu le voir durant l'épisode BRC-20 en mai 2023. Aussi, sa structure rend très complexe la confidentialité des paiements.

LIGHTNING NETWORK DAEMON (LND)

Implémentation majeure du protocole Lightning Network écrite en langage Go. Développée par Lightning Labs, LND permet la création et la gestion de canaux de paiement et de nœuds sur le réseau Lightning.

LIQUID NETWORK

Sidechain de Bitcoin conçue par Blockstream pour fournir des transactions rapides et confidentielles. Contrairement à la blockchain principale de Bitcoin, Liquid utilise un mécanisme de consensus basé sur une fédération (un groupe sélectionné d'opérateurs de nœuds, généralement des entreprises liées à Bitcoin), remplaçant ainsi le mécanisme de consensus de Nakamoto. Cette approche accélère considérablement les transactions et réduit les coûts, tout en offrant des fonctionnalités plus avancées. Liquid permet aussi l'émission d'actifs numériques, y compris des jetons représentant d'autres cryptomonnaies. Les bitcoins sur Liquid, appelés L-BTC, sont liés au BTC grâce à un système d'ancrage bilatéral reposant sur une partie de la fédération. Les participants à cette fédération sont appelés des « fonctionnaires », et il peuvent endosser à la fois le rôle de « gardien » (watchmen) et de « signataire de bloc » (blocksigner).

LITTLE-ENDIAN

Format de stockage de données dans les systèmes informatiques où les octets les moins significatifs (les « petits bouts ») sont placés en premier dans l'ordre des adresses. Dans une séquence comportant plusieurs octets, l'octet ayant le plus petit poids (par exemple, les chiffres les plus à droite en hexadécimale) est stocké en premier.

LOCK (.LOCK)

Fichier utilisé dans Bitcoin Core pour le verrouillage du répertoire de données. Il est créé lorsque bitcoind ou Bitcoin-qt démarre pour éviter que plusieurs instances du logiciel accèdent simultanément au même répertoire de données. Le but est de prévenir les conflits et les corruptions de données. Si le logiciel s'arrête de manière inattendue, le fichier .lock peut éventuellement rester et doit être supprimé manuellement avant de redémarrer Bitcoin Core.

LOGARITHME DISCRET (PROBLÈME)

Le logarithme discret est un problème mathématique qui est utilisé dans certains algorithmes cryptographiques à clé publique. Dans un groupe cyclique d'ordre q, avec un générateur g, si l'on a une équation de la forme $g^x = h$, alors x est appelé le logarithme discret de h par rapport à la base g, modulo q. En termes simples, il s'agit de déterminer l'exposant x lorsqu'on connaît g, h, et q. Le logarithme discret est donc la réciproque de l'exponentielle dans un groupe cyclique fini. Cependant, pour de grandes valeurs de q, résoudre le problème du logarithme discret est considéré comme algorithmiquement difficile. Cette propriété est exploitée pour assurer la sécurité de nombreux protocoles cryptographiques, tels que le protocole de Diffie-Hellman pour l'échange de clés. Le logarithme discret est aussi utilisé dans la cryptographie à courbes elliptiques (ECC), entre autres dans l'algorithme ECDSA (Elliptic Curve Digital Signature Algorithm). Dans le contexte des courbes elliptiques, le problème du logarithme discret s'étend à la recherche d'un scalaire k tel que $k \cdot G = K$, où G et K sont des points sur la courbe, et $\cdot$ représente l'opération de multiplication de points. Dans le contexte de

Bitcoin, les transactions standards utilisent soit ECDSA, soit le protocole de Schnorr, afin de bloquer des UTXO. Ils reposent tous deux sur l'impossibilité de calculer le logarithme discret.

LOOP

Service développé par Lightning Labs conçu pour faciliter l'équilibrage de liquidités dans les canaux Lightning. Loop permet aux utilisateurs de transférer des fonds entre Bitcoin et le Lightning Network, sans avoir à fermer ou ouvrir un canal. Loop aide ainsi à optimiser sa liquidité et à réduire les frais de gestion de ses canaux.

M

MAGICAL BITCOIN

Ancien nom de la collection d'outils et de bibliothèques pour développeurs BDK.

*Pour plus d'informations, voir la définition de **BDK (BITCOIN DEV KIT)**.*

MAGIC NETWORK

Constantes utilisées dans le protocole Bitcoin pour identifier le réseau spécifique (mainnet, testnet, regtest…) d'un message échangé entre nœuds. Ces valeurs sont inscrites au début de chaque message pour faciliter l'identification des messages liés à Bitcoin dans le flux de données. Les Magic Network sont conçus pour être rarement présents dans des données de communication ordinaires. En effet, ces 4 octets sont peu fréquents dans l'ASCII, sont invalides en UTF-8 et génèrent un très grand entier de 32 bits, peu importe le format de stockage des données. Les Magic Network sont (en format little-endian) :

- Mainnet : `f9beb4d9` ;
- Testnet : `0b110907` ;
- Regtest : `fabfb5da` .

C'est 4 octets sont parfois également nommés « Magic Number », « Magic Bytes » ou encore « Start String ».

MAJORITÉ ÉCONOMIQUE

Désigne la plus grande proportion de l'activité économique liée à la monnaie bitcoin, contrôlée par les commerçants. Un commerçant désigne toute entité physique ou morale acceptant d'échanger un bien ou un service contre du BTC. Ces commerçants, qui incluent les commerces, les utilisateurs, les plateformes d'échange, et les mineurs, varient en taille et en influence économique. Certains sont des acteurs majeurs, générant une activité économique substantielle, tandis que d'autres sont plus modestes. La majorité économique est donc définie par ceux dont l'activité économique combinée représente la part prépondérante sur cette monnaie. Cette majorité a une influence sur les règles de consensus, notamment en cas de fork.

MALLÉABILITÉ (TRANSACTION)

Se réfère à la possibilité de modifier légèrement la structure d'une transaction Bitcoin, sans en altérer l'effet, mais tout en changeant l'identifiant de transaction (TxID). Cette propriété peut être exploitée malicieusement pour induire en erreur les parties prenantes sur le statut d'une transaction, causant ainsi des problèmes comme la double dépense. La malléabilité était rendue possible par la flexibilité de la transaction électronique utilisée. Le soft fork SegWit a notamment été introduit pour empêcher cette malléabilité des transactions Bitcoin, rendant compliquée une implémentation du Lightning Network. Il y parvient en écartant les données malléables de la transaction du calcul du TxID.

Bien que ce soit rare, on retrouve parfois le terme de « mutabilité » pour évoquer le même concept.

MAN-IN-THE-MIDDLE (MITM)

Attaque dans laquelle un acteur malveillant se place clandestinement entre deux parties communiquant, interceptant et potentiellement modifiant les messages échangés, sans que les deux parties ne remarquent sa présence.

En français, on parle d'une « attaque de l'homme du milieu » ou « HDM ».

MAPPER (TO MAP)

Dans le contexte de l'informatique, mapper désigne le processus d'associer des éléments d'un ensemble de données à des éléments d'un autre ensemble de données de manière systématique. Cette association permet aux données du premier ensemble de se substituer à celles du second ensemble ou de transitionner de l'un à l'autre. Cette technique est souvent utilisée dans les opérations de transformation de données.

MASF (MINER-ACTIVATED SOFT FORK)

Qualifie un soft fork dans Bitcoin lorsque son activation provient d'une action des mineurs. Les MASF sont une famille de méthodes d'activation de soft fork sur Bitcoin. Dans ces approches, les mineurs signalent leur accord et leur préparation pour une mise à jour du protocole en minant des blocs qui soutiennent le verrouillage du soft fork. Si une majorité significative de mineurs se prononce en faveur du soft fork, la mise à jour est considérée comme acceptée et est activée ultérieurement. Ce processus permet d'éviter la division de la blockchain et de maintenir l'unité du réseau. Le MASF est préféré pour son approche plus douce et consensuelle, réduisant le risque de scission de la blockchain tout en assurant que la majorité de la puissance de calcul soutient la nouvelle mise à jour. Les méthodes d'activation BIP34, BIP9, BIP8 (si `LOT=false` ou si le seuil de vote est atteint) ou encore Speedy Trial sont des MASF.

MAST (MERKELISED ALTERNATIVE SCRIPT TREE)

Technique employant un arbre de Merkle pour résumer un nombre arbitraire de conditions de dépenses sélectionnées par l'utilisateur dans une adresse de réception, dont une doit être remplie pour dépenser les bitcoins concernés. L'utilisation d'un arbre de Merkle permet à l'utilisateur de choisir quelle condition il souhaite remplir sans révéler les détails des autres conditions sur la blockchain. Cela permet de réduire les frais liés à ces scripts, de créer des conditions beaucoup plus lourdes et, sur un temps plus long, d'améliorer la confidentialité de l'utilisateur (en plus de l'utilisation conjointe de Schnorr). Ce concept a fait l'objet de plusieurs propositions, mais il a finalement était ajouté à Bitcoin via le soft fork Taproot en 2021.

> *Initialement, « MAST » était l'acronyme de « Merklized Abstract Syntax Tree ». L'utilisation qui en est faite dans le cadre de Taproot n'a plus rien à voir avec un « Abstract Syntax Tree ». Toutefois, les utilisateurs continuaient d'employer ce terme de MAST. Anthony Towns a donc proposé de modifier la signification initiale tout en conservant cet acronyme largement employé avec : « Merklized Alternative Script Trees ».*

MASTER FINGERPRINT

Empreinte de 4 octets (32 bits) de la clé privée maîtresse dans un portefeuille hiérarchique déterministe (HD). Elle est obtenue en calculant le hash `SHA256` de la clé privée maîtresse, suivi d'un hash `RIPEMD160`, procédé désigné par `HASH160` sur Bitcoin. La Master Fingerprint sert à identifier un portefeuille HD, indépendamment des chemins de dérivation, mais en prenant en compte la présence ou non d'une passphrase. C'est une information concise qui permet de faire référence à l'origine d'un ensemble de clés, sans pour autant dévoiler des informations sensibles sur le portefeuille.

MAX_BLOC_SIZE

Constante qui spécifie la taille maximale qu'un bloc peut avoir sur Bitcoin. Historiquement, cette limite était fixée à 1 Mo, une mesure mise en place par Satoshi Nakamoto en 2010 afin de prévenir le spam et de maintenir une certaine décentralisation du réseau.

MEMPOOL

Contraction des termes « memory » et « pool ». Cela désigne un espace virtuel dans lequel les transactions Bitcoin en attente d'inclusion dans un bloc sont regroupées. Lorsqu'une transaction est créée et diffusée sur le réseau Bitcoin, elle est d'abord vérifiée par les nœuds du réseau. Si elle est considérée comme valide, elle est alors placée dans la Mempool, où elle reste jusqu'à ce qu'elle soit sélectionnée par un mineur pour être incluse dans un bloc. Il est important de noter que chaque nœud du réseau Bitcoin maintient sa propre Mempool, et donc, il peut y avoir des variations dans le contenu de la Mempool entre différents nœuds à un moment donné. Notamment, le paramètre `maxmempool` dans le fichier `bitcoin.conf` de chaque nœud permet aux opérateurs de contrôler la quantité de RAM (mémoire vive) que leur nœud utilisera pour stocker les transactions en attente dans la Mempool. En limitant la taille de la Mempool, les opérateurs de nœuds peuvent éviter que celle-ci ne consomme trop de ressources sur leur système. Ce paramètre est spécifié en mégaoctets (MB). La valeur par défaut de Bitcoin Core à ce jour est de 300 Mo. Les transactions présentent dans les mempool sont provisoires. Elles ne doivent pas être considérées comme immuable tant qu'elles ne sont pas incluses dans un bloc, et après un certain nombre de confirmations. Celles-ci peuvent souvent être remplacées, purgées ou double-dépensées.

MEMPOOL.DAT

Nom du fichier de données utilisé par le logiciel Bitcoin Core pour stocker l'état actuel de la mempool, qui est l'ensemble des transactions non confirmées en attente d'être ajoutées à un bloc.

MERGE

Dans le cadre de Git, représente l'action d'intégrer les modifications d'une branche à une autre, typiquement de ramener les développements d'une branche secondaire dans la branche principale. Cette opération permet de combiner les historiques de commit des branches concernées et de résoudre les éventuels conflits pour maintenir l'intégrité du logiciel.

En français, on peut traduire « merge » par « fusion ».

MERKLE BLOCK

Structure de données utilisée dans le cadre du BIP37 (*Transaction Bloom Filtering*) pour fournir une preuve compacte de l'inclusion de transactions spécifiques dans un bloc. C'est notamment utilisé pour les portefeuilles SPV. Le Merkle Block contient les en-têtes de bloc, les transactions filtrées et un arbre de Merkle partiel, permettant aux clients légers de vérifier rapidement si une transaction appartient à un bloc sans télécharger toutes les transactions.

MÉTHODE D'ACTIVATION

Processus par lequel la communauté d'utilisateurs décide de l'implémentation d'un soft fork sur le protocole Bitcoin, en cherchant à éviter une séparation de la blockchain. Ce processus consiste à solliciter l'opinion des mineurs pour approuver un soft fork avant son activation. Si une majorité

importante accepte le soft fork, le risque de scission de la blockchain est minimisé. Ce consensus est crucial car, si une majorité de mineurs refusent de faire la modification, le soft fork pourrait créer deux chaînes distinctes - une avec les règles modifiées et l'autre sans. Il existe 2 grandes catégories de méthodes d'activation :

- Les UASF (*User-Activated Soft Fork*) lorsque ce sont les nœuds qui imposent la mise à jour ;
- Les MASF (*Miner-Activated Soft Fork*) lorsque ce sont les mineurs qui déclenchent l'activation.

Il existe de nombreuses méthodes d'activation différentes qui ont été testées au fur et à mesure de l'évolution de Bitcoin. À l'époque de Satoshi, le processus d'activation n'était pas formellement établi. Les modifications étaient souvent arbitraires et parfois même réalisées sans informer la communauté. Plus tard, la méthode du « Flag Day » a été adoptée. Après le retrait de Satoshi, d'autres méthodes ont été successivement utilisées, notamment le BIP34, le BIP9, le BIP8, et enfin le Speedy Trial.

MÉTHODE GÉOMÉTRIQUE

Méthode de calcul de la rémunération des mineurs dans le contexte des pools de minage. Ce système de paiement est basé sur un score, conçu pour contrer le phénomène de pool hopping. Elle assure que le paiement par part soumise reste constant, indépendamment du moment de soumission. Les mineurs accumulent des scores, calculés avec un facteur de décroissance, et les paiements sont calculés à la fin du cycle. Ils sont proportionnels à leur score. Cette méthode implique des frais variables et fixes pour le mineur, et réduit la variance des paiements par part.

MINAGE

Action de participer à la preuve de travail (Proof-of-Work) du système Bitcoin. La preuve de travail est un mécanisme de résistance aux attaques Sybil. Elle est à la base du protocole de consensus de Nakamoto, qui est le principe utilisé pour établir un accord sur une version unique du registre distribué entre les différents nœuds du réseau. Concrètement, le minage est la recherche d'une valeur qui, une fois passée dans une fonction mathématique aléatoire, donne un résultat inférieur à un nombre cible. Cette cible de la preuve de travail est ajustée tous les 2016 blocs par les nœuds. C'est ce que l'on appelle l'ajustement de la difficulté. On abaisse le nombre cible pour augmenter la difficulté de minage, ou on l'augmente pour baisser la difficulté, en fonction de l'évolution de la puissance de calcul déployée par les mineurs durant la période précédente. Ce travail effectué par les mineurs est récompensé à chaque bloc valide trouvé. Le mineur gagnant empoche une récompense pécuniaire, composée de la subvention de bloc (création de nouveaux bitcoins ex-nihilo), et des frais de transaction. Aujourd'hui, la difficulté de la preuve de travail sur Bitcoin est telle que le minage nécessite une grande puissance de calcul pour parvenir à gagner des blocs. En conséquence, il faut souvent disposer de puces électroniques spécialisées dans l'exécution de `SHA256`, c'est ce que l'on appelle un ASIC, et participer dans des pools de minage.

MINAGE ÉGOÏSTE

Voir la définition de **SELFISH MINING**.

MINAGE FUSIONNÉ

Technique de consensus de sidechain permettant aux mineurs de Bitcoin de travailler simultanément sur la chaîne principale et sur une ou plusieurs sidechains, sans pour autant devoir fournir plus de travail de calcul. Il s'agit donc de réutiliser la preuve de travail de Bitcoin pour des applications tierces. Toutefois, le minage fusionné présente un désavantage notable pour le mineur : il nécessite

l'installation et l'exécution d'un logiciel de nœud spécifique à chaque sidechain pour permettre la réutilisation de ses preuves de travail. De plus, la récompense obtenue pour le minage d'une sidechain est versée sur celle-ci et non directement en BTC sur la blockchain principale.

En anglais, on parle de « Merged Mining » ou « MM ».

MINAGE FUSIONNÉ AVEUGLE

Technique de consensus de sidechain permettant aux mineurs de Bitcoin de travailler simultanément sur la chaîne principale et sur une ou plusieurs sidechains, sans pour autant devoir fournir plus de travail de calcul. Contrairement au minage fusionné classique, cette méthode ne nécessite pas de configurer un nouveau nœud pour chaque sidechain exploitée. Dans le cadre du Blind Merged Mining (BMM), chaque sidechain est gérée par des opérateurs de nœud indépendants, responsables de la création des blocs et de la récolte des récompenses associées sur la sidechain. En contrepartie, ces opérateurs doivent acheter des preuves de travail auprès des mineurs de la blockchain principale pour valider leurs blocs sur la sidechain. Ainsi, les mineurs de Bitcoin reçoivent leurs récompenses du minage fusionné des sidechains en BTC, directement sur la chaîne principale. Cette méthode, développée par Paul Sztorc pour les drivechains, nécessite l'implémentation du BIP301 pour être opérationnelle sur le réseau Bitcoin.

En anglais, on parle de « Blind Merged Mining » ou « BMM ».

MINEUR

Dans le contexte de Bitcoin, un mineur fait référence à un ordinateur engagé dans le processus de minage, qui consiste à participer à la preuve de travail (Proof-of-Work). Le mineur regroupe les transactions en attente dans sa mempool pour former un bloc candidat. Ensuite, il recherche un hachage valide, inférieur ou égal à la cible, pour l'entête de ce bloc en modifiant les différents nonces. S'il trouve un hachage valide, il diffuse son bloc au réseau Bitcoin et empoche la récompense pécuniaire associée, composée de la subvention de bloc (création de nouveaux bitcoins ex-nihilo), et des frais de transaction. Par extension, le terme de « mineur » désigne également la personne ou l'entité qui possède et opère un ou plusieurs de ces ordinateurs.

MINISCRIPT

Framework permettant de fournir un cadre pour programmer des scripts de manière sécurisée sur Bitcoin. Le langage natif de Bitcoin s'appelle script. Celui-ci est assez complexe à utiliser en pratique, notamment pour des applications sophistiquées et personnalisées. Notamment, il est très difficile de vérifier les limitations d'un script. Miniscript utilise un sous-ensemble de scripts Bitcoin pour simplifier leur création, leur analyse et leur vérification. Chaque miniscript est équivalent 1 pour 1 avec un script natif. On utilise un langage de Policies facile à utiliser, qui est ensuite compilé en Miniscript, pour enfin correspondre à un Script natif. Miniscript permet ainsi aux développeurs de construire des scripts sophistiqués d'une manière plus sûre et plus fiable. Les propriétés essentielles de Miniscript sont les suivantes :

- Il permet une analyse statique du script, notamment des conditions de dépenses qu'il permet et de son coût en termes de ressources ;
- Il permet de réaliser des scripts qui respectent le Consensus ;
- Il permet d'analyser si oui ou non, les différents chemins de dépense respectent les règles standards des nœuds ;

- Il permet de mettre en place un standard général, compréhensible et composable, pour l'ensemble des logiciels et matériels de portefeuille.

Le projet Miniscript a été lancé en 2018 par Peter Wuille, Andrew Poelstra et Sanket Kanjalkar, via l'entreprise Blockstream. Miniscript est ajouté au wallet Bitcoin Core en mode watch-only en décembre 2022 avec la version 24.0, puis complètement en mai 2023 avec la version 25.0.

MINITAPSCRIPT

Version de Miniscript pour Tapscript. Tapscript dispose de quelques différences notables avec Script dans sa version originale. MiniTapscript fournit ainsi la prise en charge de Tapscript dans Miniscript.

> *Ce terme est parfois contesté. En effet, certains bitcoiners préfèrent parler de « TapMiniscript ». Pour plus d'informations, voir la définition de **MINISCRIPT** et de **TAPSCRIPT**.*

MODÈLE DE SCRIPT

Template permettant l'utilisation de scripts standards. Un modèle de script est essentiellement une petite liste d'OPcodes mis ensembles pour former une norme qui spécifie une manière d'établir des conditions de dépenses sur des bitcoins. Exemples de modèles de script : P2PK, P2PKH, P2WPKH, P2SH…

MTP (MEDIAN TIME PAST)

Concept utilisé dans le protocole Bitcoin pour déterminer une marge sur l'horodatage consensuel du réseau. Le MTP est défini comme la médiane des horodatages des 11 derniers blocs minés. L'utilisation de cet indicateur permet d'éviter les désaccords entre les nœuds sur l'heure exacte en cas de décalage. Le MTP était initialement utilisé pour vérifier la validité de l'horodatage des blocs par rapport au passé. Depuis le BIP113, il est également utilisé comme référentiel du temps du réseau pour vérifier la validité des opérations de verrouillages temporels (`nLockTime`, `OP_CHECKLOCKTIMEVERIFY`, `nSequence` et `OP_CHECKSEQUENCEVERIFY`).

N

NESTED SEGWIT

Standard de scripts utilisés pour envelopper des scripts SegWit natifs, au sein d'un script P2SH. Les scripts Nested SegWit ont été inventé au lancement de SegWit pour faciliter son adoption. Ils permettent d'utiliser ce nouveau standard, même sur des wallets pas encore compatibles nativement avec SegWit. C'est une sorte de script de transition vers la nouvelle norme. Aujourd'hui, il n'est donc plus très pertinent d'utiliser ce type de scripts SegWit wrappés, puisque la plupart des wallets ont implémenté du SegWit natif.

*Pour plus d'informations, voir la définition de **P2SH-P2WPKH**.*

NETWORK-ADJUSTED TIME (NAT)

Estimation du temps universel basée sur les horloges des nœuds du réseau. Chaque nœud calcule son NAT en prenant la médiane des différences de temps entre son horloge locale (UTC) et celles des nœuds avec lesquels il est connecté, puis en additionnant son horloge locale avec la médiane de ces différences, jusqu'à un maximum de 70 minutes. Le network-adjusted time est donc une médiane du temps des nœuds calculée en local par chaque nœud. Ce référentiel est ensuite utilisé pour vérifier la validité des horodatages des blocs. En effet, pour qu'un bloc soit accepté par un nœud, son horodatage doit se situer entre le MTP (temps médian des 11 derniers blocs minés) et le NAT plus 2 heures : `MTP < Horodatage valide < (NAT + 2h)`.

NEW YORK AGREEMENT (NYA)

Réunion privée qui s'est tenue en 2017, rassemblant plus de 50 entreprises de l'écosystème Bitcoin, à la suite de la conférence Consensus 2017. L'objectif de cette réunion était de finir le débat de longue date sur le passage à l'échelle de Bitcoin en parvenant à un accord. De cette rencontre est née la proposition SegWit2x, s'inspirant de la précédente proposition SegWit2Mb. Elle prévoyait deux modifications majeures du protocole Bitcoin :

- L'adoption de SegWit avec un seuil d'activation fixé à 80
- Un hard fork destiné à augmenter la taille maximale des blocs de 1 Mo à 2 Mo.

Malgré le signalement positif de plus de 80 % des mineurs, le projet n'a pas su rallier un consensus suffisant, aboutissant à son abandon. Cet événement a été interprété par de nombreux utilisateurs et développeurs comme une tentative d'attaque de Bitcoin.

NLOCKTIME

Champ intégré dans les transactions qui définit une condition temporelle avant laquelle la transaction ne peut être ajoutée à un bloc valide. Ce paramètre permet de spécifier un temps précis (timestamp Unix) ou un nombre de blocs spécifique comme condition pour que la transaction soit considérée comme valide. C'est donc un timelock absolu (pas relatif). Le `nLockTime` agit sur l'intégralité de la transaction et permet effectivement de vérifier le temps, alors que l'opcode `OP_CHECKLOCKTIMEVERIFY` permet uniquement de comparer la valeur en haut de la pile avec la valeur du `nLockTime`.

*Pour plus d'informations, voir la définition de **OP_CHECKLOCKTIMEVERIFY** et **TIMELOCK**.*

NŒUD

Dans le réseau Bitcoin, un nœud (ou « node » en anglais) est un ordinateur qui exécute un client du protocole Bitcoin (comme Bitcoin Core par exemple). Il participe au réseau en maintenant une copie

de la blockchain, en relayant et en vérifiant les transactions et les nouveaux blocs et, optionnellement, en participant au processus de minage. La somme de tous les nœuds Bitcoin représente le réseau Bitcoin en lui-même. Il existe plusieurs types de nœuds sur Bitcoin, dont les plus notables sont les nœuds complets et les nœuds légers. Les nœuds complets conservent une copie intégrale de la blockchain, vérifient toutes les transactions et les blocs selon les règles de consensus, et participent activement à la diffusion de transactions et de blocs sur le réseau. En revanche, les nœuds légers, ou nœuds SPV (Simple Payment Verification), ne conservent qu'une partie de la blockchain et comptent sur les nœuds complets pour obtenir des informations sur les transactions.

> *Certains différencient également les nœuds dits « élagués » (« pruned node » en anglais). Ce sont des nœuds complets, qui téléchargent et vérifient tous les blocs depuis le bloc Genesis, mais qui ne conserve que les blocs les plus récents en mémoire.*

NŒUD COMPLET

Un nœud complet, ou « Full Node » en anglais, fait référence à un ordinateur qui exécute un client du protocole Bitcoin, et qui télécharge, vérifie et stocke la totalité de la blockchain, soit l'historique complet des transactions depuis le bloc Genesis. Un nœud complet vérifie de manière autonome toutes les transactions et les blocs en fonction des règles de consensus de Bitcoin. C'est donc ce type de nœud qui procure le plus haut niveau de vérification pour son utilisateur, et qui permet de réduire au maximum le besoin de confiance envers une tierce partie. Le nœud complet nécessite plus de ressources de stockage, de puissance de calcul, de RAM et de bande passante qu'un nœud léger (SPV).

NŒUD ÉLAGUÉ

Un nœud élagué, en anglais « Pruned Node », est un nœud complet qui exécute un élagage de la blockchain. Cela consiste à supprimer de manière progressive les blocs les plus anciens, après les avoir dûment vérifiés, pour conserver seulement les blocs les plus récents. La limite de conservation est renseignée dans le fichier `bitcoin.conf` via le paramètre `prune=n`, ou `n` est la taille maximale prise par les blocs en mégaoctets (Mo). Si `0` est noté après ce paramètre, alors l'élagage est désactivé, et le nœud conserve la blockchain dans son intégralité. Les nœuds élagués sont parfois considérés comme des types de nœuds différents des nœuds complets. L'utilisation d'un nœud élagué peut s'avérer particulièrement intéressante pour les utilisateurs confrontés à des contraintes en termes de capacité de stockage. Actuellement, un nœud complet doit disposer d'environ 500 Go pour le stockage de la blockchain. Un nœud élagué peut limiter le stockage requis jusqu'à 550 Mo. Bien qu'il utilise moins d'espace disque, un nœud élagué maintient un niveau de vérification et de validation semblable à celui d'un nœud complet. Les nœuds élagués offrent donc plus de confiance à leurs utilisateurs en comparaison avec les nœuds légers (SPV).

NŒUD SPV (OU NŒUD LÉGER)

Un nœud SPV (Simple Payment Verification), parfois nommé « nœud léger », est un client léger d'un nœud Bitcoin qui permet aux utilisateurs de valider les transactions sans avoir à stocker l'intégralité de la blockchain. Au lieu de cela, un nœud SPV stocke seulement les entêtes des blocs, et obtient des informations sur des transactions spécifiques en interrogeant des nœuds complets lorsque nécessaire. Ce principe de vérification est rendu possible par la structure des transactions dans les blocs Bitcoin, qui sont organisées au sein d'un accumulateur cryptographique (Arbre de Merkle). Cette approche est avantageuse pour les appareils avec des ressources limitées, tels que les téléphones portables. Cependant, les nœuds SPV font confiance aux nœuds complets pour la disponibilité des informations, ce qui implique un niveau de confiance supplémentaire et, par conséquent, une moindre

sécurité par rapport aux nœuds complets. Les nœuds SPV ne peuvent pas valider les transactions de manière autonome, mais ils peuvent vérifier si une transaction est incluse dans un bloc en consultant les preuves de Merkle.

NONCE

Dans le contexte de l'informatique, le terme « nonce » désigne un nombre utilisé seulement une seule fois, puis remplacé. Il est souvent aléatoire ou pseudo-aléatoire. On l'utilise dans divers protocoles cryptographiques pour garantir la sécurité du procédé. Par exemple, les signatures ECDSA utilisées au sein du protocole Bitcoin incluent l'utilisation d'un nonce. Cela veut dire que ce nombre doit être nouveau pour chaque signature. Si ce n'est pas le cas, il est possible de calculer la clé privée utilisée en rapprochant deux signatures qui utilisent le même Nonce. On utilise également des nonces dans le processus de minage sur Bitcoin. Les mineurs incrémentent ces valeurs modifiables au sein de leurs blocs candidats. Ils modifient la valeur du nonce dans le but de trouver une empreinte cryptographique inférieure ou égale à la cible de difficulté. Ce processus nécessite une grande puissance de calcul, car il s'agit d'une recherche exhaustive parmi un grand nombre de nonces possibles. Lorsqu'un mineur trouve un nonce qui, lorsqu'il est inclus dans son bloc, produit un condensat répondant aux critères de difficulté, le bloc est diffusé au réseau, et le mineur remporte la récompense.

> *En 2010, des chercheurs ont découvert que la PlayStation 3 de Sony utilisait le même nonce lors de la signature de différents paquets de code. Cette réutilisation du nonce a permis aux attaquants de calculer la clé privée utilisée pour signer le logiciel. Avec la clé privée en main, les attaquants pouvaient créer des signatures valides pour n'importe quel code, ce qui leur permettait d'exécuter des logiciels non autorisés, y compris des jeux piratés ou des systèmes d'exploitation personnalisés, directement sur la PS3.*

NSEQUENCE

Le champ `nSequence` dans une entrée de transaction Bitcoin est utilisé pour indiquer la manière dont cette entrée est verrouillée dans le temps. À l'origine, il visait à permettre le remplacement dynamique de transactions dans les mempools afin de permettre un système de paiement en surcouche similaire à Lightning. Toutefois, son utilisation a évolué avec l'introduction du timelock relatif via le BIP68. Le champ `nSequence` peut désormais spécifier un délai relatif avant qu'une transaction soit incluse dans un bloc. Ce délai peut être défini en terme de nombre de bloc, ou bien comme un multiple de 512 secondes (c'est-à-dire, du temps réel). Notons que cette nouvelle interprétation du champs `nSequence` est uniquement valide si le champs `nVersion` est supérieur ou égal à `2`. Cette interprétation du champs `nSequence` se fait au niveau des règles de consensus de Bitcoin. Par ailleurs, au niveau des règles de standardisation, ce champ est également utilisé pour le signalement de RBF. Si une transaction inclue un `nSequence` inférieur à `0xfffffffe`, alors elle pourra être remplacée via RBF sur les nœuds qui suivent cette politique.

> *Pour plus d'informations, voir la définition de **OP_CHECKSEQUENCEVERIFY** et **TIMELOCK**.*

NULL DATA

Type de transaction Bitcoin qui permet d'insérer une petite quantité de données arbitraires grâce à un `OP_RETURN`. Les bitcoins éventuellement associés à ce type d'output sont non dépensables de manière prouvée, car l'`OP_RETURN` signale un script invalide.

NULLDUMMY

Règle de consensus introduite avec le BIP147 dans le soft fork SegWit qui exige que l'élément factice (« *dummy element* ») utilisé dans les opcodes `OP_CHECKMULTISIG` et `OP_CHECKMULTISIGVERIFY` soit un tableau d'octets vide (`OP_0`). Cette mesure a été mise en place pour éliminer un vecteur de malléabilité en interdisant toute valeur autre que `OP_0` pour cet élément.

*Pour plus d'informations, voir la définition de **DUMMY ELEMENT** et **BIP147**.*

NVERSION

Le champ `nVersion` dans une transaction Bitcoin sert à indiquer la version du format de transaction utilisé. Il permet au réseau de distinguer les différentes évolutions du format de transaction au fil du temps, et d'appliquer les règles correspondantes. Ce champ n'a aucun impact au niveau des règles de consensus. Cela signifie que toute valeur attribuée à ce champ n'entraîne pas l'invalidation de la transaction. En revanche, le champ `nVersion` dispose de règles de standardisation qui n'acceptent que la valeur de `1` et de `2` actuellement. Pour le moment, ce champ est seulement utile pour l'activation du champ `nSequence`.

*Pour plus d'informations, voir la définition de **NSEQUENCE**.*

O

OBOE (OFF-BY-ONE ERROR)

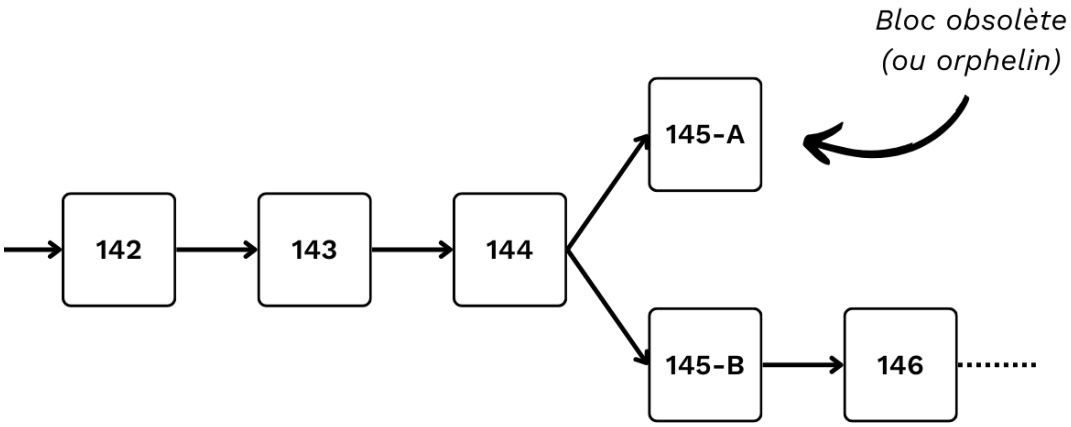
Erreur de logique où une boucle itère une fois de trop ou de moins, souvent due à une mauvaise utilisation des opérateurs de comparaison ou de mauvais indices dans la gestion des structures de données. Dans le contexte de Bitcoin, on retrouve ce bug dans le cas du « *dummy element* » dans `OP_CHECKMULTISIG`, où un élément supplémentaire est consommé par erreur.

> *En français, on peut traduire ce terme par « erreur de décalage unitaire ». Pour plus d'informations, voir les définitions de **DUMMY ELEMENT** et **BIP147**.*

OBSOLÈTE (BLOC)

Fait référence à un bloc sans enfant : un bloc valide mais exclu de la chaîne principale de Bitcoin. Il se produit lorsque deux mineurs trouvent un bloc valide sur une même hauteur de chaîne durant un court laps de temps et le diffusent sur le réseau. Les nœuds finissent par choisir un seul bloc à inclure dans la chaîne, selon le principe de la chaîne avec le plus de quantité de travail accumulé, rendant l'autre « orphelin ». Le processus menant à la production d'un bloc obsolète est le suivant :

- Deux mineurs trouvent un bloc valide à une même hauteur de chaîne durant un court intervalle de temps. Nommons les `Bloc A` et `Bloc B` ;
- Chacun diffuse son bloc au réseau de nœuds Bitcoin. Chaque nœud dispose maintenant de 2 blocs à une même hauteur. Il existe donc deux chaînes valides ;
- Les mineurs continuent de chercher des preuves de travail pour les blocs suivants, mais pour ce faire, ils doivent obligatoirement choisir un seul bloc entre le `Bloc A` et le `Bloc B` au-dessus duquel ils vont miner ;
- Un mineur trouve finalement un bloc valide au-dessus du `Bloc B` . Appelons le `Bloc B+1` ;
- Il diffuse `Bloc B+1` aux nœuds du réseau ;
- Puisque les nœuds suivent la chaîne la plus longue (avec le plus de quantité de travail accumulé), ils vont estimer que la `Chaîne B` est celle qu'il faut suivre ;
- Ils vont donc abandonner le `Bloc A` qui ne fait plus partie de la chaîne principale. Il est donc devenu un bloc obsolète.



En anglais, on parle de « Stale Block ». En français, on peut également dire « bloc périmé » ou « bloc abandonné ». Même si je ne suis pas en accord avec cet usage, certains bitcoiners utilisent le terme de « bloc orphelin » pour désigner ce qui est en réalité un bloc obsolète. Pour plus d'informations, voir la définition de ***ORPHELIN (BLOC)****.*

OCTET (BYTE)

Unité de mesure de données informatiques équivalant à 8 bits. Chaque bit est un chiffre binaire (0 ou 1), ce qui signifie qu'un octet peut représenter 256 (2^8) combinaisons uniques.

OFFCHAIN

Fait référence aux transactions ou activités plus ou moins liées à Bitcoin qui se produisent en dehors de la blockchain principale, mais qui disposent d'un lien ou d'un ancrage avec celle-ci. Elles ne sont pas immédiatement enregistrées sur la blockchain, mais nécessitent des mécanismes supplémentaires pour assurer leur sécurité et leur finalité. Ces opérations se justifient souvent par un désir d'outre-passer les limitations techniques inhérentes à Bitcoin afin de disposer de transactions à finalité rapide, à bas frais, avec plus de capacité ou de fonctionnalités.

ONCHAIN

Désigne les transactions enregistrées directement sur la blockchain Bitcoin. Ce terme s'oppose à « offchain » qui désigne des opérations ayant un rapport plus ou moins prononcé avec la blockchain Bitcoin, mais qui se déroulent en dehors de la blockchain principale.

ONION_PRIVATE_KEY

Fichier anciennement utilisé dans Bitcoin Core pour stocker une clé privée associée à un service caché Tor V2 pour l'option `-listenonion`. Ce fichier n'est plus utilisé depuis la version 0.21.0 au profit de la V3 de Tor.

ONION_V3_PRIVATE_KEY

Fichier utilisé dans Bitcoin Core pour stocker une clé privée associée à un service caché Tor pour l'option `-listenonion`. Lorsque cette option est activée, bitcoind crée automatiquement un service caché Tor, permettant au nœud de communiquer sur le réseau Tor.

OP_0 (0X00)

Pousse la valeur 0 sur la pile. Il est souvent utilisé pour représenter la valeur booléenne `faux` dans les scripts. OP_0 est également utilisé pour initialiser les scripts.

*OP_0 est identique à **OP_FALSE** et **OP_PUSHNUM_0**.*

OP_0NOTEQUAL (0X92)

Vérifie si l'élément au sommet de la pile est différent de zéro. Si l'élément est autre que zéro, il pousse `1` (`vrai`) sur la pile, sinon, il pousse `0` (`faux`).

OP_1 (0X51)

Pousse la valeur `1` sur la pile. Il est souvent utilisé pour représenter la valeur booléenne `vrai` dans les scripts.

*OP_1 est identique à **OP_TRUE** et **OP_PUSHNUM_1**.*

OP_1ADD (0X8B)

Ajoute `1` à la valeur en haut de la pile.

OP_1NEGATE (0X4F)

Pousse la valeur `-1` sur la pile. Cet opcode est utilisé dans les scripts pour représenter la valeur négative `-1`.

OP_1SUB (0X8C)

Soustrait `1` à la valeur en haut de la pile.

OP_2 à OP_16 (0X52 à 0X60)

Les opcodes de OP_2 jusqu'à OP_16 poussent les valeurs numériques respectives de 2 à 16 sur la pile. On les utilise pour simplifier les scripts en permettant l'insertion de petites valeurs numériques. Ce type d'opcode est notamment utilisé dans les scripts multisignatures. Voici un exemple de `ScriptPubKey` pour un multisig 2/3 : `OP_2 Clé publique A Clé publique B Clé publique C OP_3 OP_CHECKMULTISIG`

*Tous ces opcodes sont parfois également nommés **OP_PUSHNUM_N**.*

OP_2DROP (0XD6)

Supprime les deux éléments en haut de la pile. Autrement dit, OP_2DROP supprime le sommet de la pile et le deuxième élément de la pile. Cet opcode est l'équivalent de l'enchainement de deux OP_DROP.

OP_2DUP (0X6E)

Duplique les deux éléments en haut de la pile, puis les place en haut de la pile. Par exemple, si la pile est D C B A, OP_2DUP produira : D C B A B A.

OP_2OVER (0X70)

Copie les deux éléments qui se trouvent à la quatrième et à la troisième place en partant du haut de la pile, puis les place en haut de la pile. Par exemple, si la pile est D C B A, OP_2OVER produira : D C B A D C.

OP_2ROT (0X71)

Déplace les deux éléments qui se trouvent à la sixième et à la cinquième place du sommet de la pile vers le sommet. Par exemple, si la pile est F E D C B A, OP_2ROT produira : D C B A F E.

OP_2SWAP (0X72)

Échange les deux éléments situés au sommet de la pile avec les deux éléments situés juste en dessous d'eux. Par exemple, si la pile est D C B A, OP_2SWAP produira : B A D C.

OP_3DUP (0X6F)

Duplique les trois éléments en haut de la pile, puis les place en haut de la pile. Par exemple, si la pile est D C B A, OP_3DUP produira : D C B A C B A.

OP_ABS (0X90)

Remplace l'élément supérieur de la pile par sa valeur absolue. Cette opération supprime le signe de l'élément, transformant toute valeur négative en positive, sans changer les valeurs positives.

OP_ADD (0X93)

Additionne les deux éléments au sommet de la pile. Il prend les deux valeurs en haut de la pile, il les additionne et il les remplace par le résultat.

OP_BOOLAND (0X9A)

Reproduit le comportement d'une porte logique AND. Il prend deux valeurs au sommet de la pile et renvoie 1 seulement si les deux valeurs sont non nulles. Dans le cas contraire, il renvoie 0.

OP_BOOLOR (0X9B)

Reproduit le comportement d'une porte logique OR. Il prend deux valeurs au sommet de la pile et renvoie 1 si l'un ou l'autre des éléments ou les deux sont non nuls. Dans le cas contraire, il renvoie 0.

OP_CAT (0X7E)

Permet de concaténer les deux éléments en haut de la pile (c'est-à-dire de les mettre bout-à-bout). Cet opcode a été désactivé, il est donc actuellement impossible de l'utiliser. Toutefois, il est récemment revenu sur le devant de la scène. Certains souhaiteraient pouvoir l'ajouter à Tapscript afin de permettre la combinaison d'objets sur la pile et ainsi améliorer l'expressivité de ce langage. Ce simple outil supplémentaire pourrait permettre :

- L'utilisation des signatures de Lamport qui sont à priori résistantes aux attaques quantiques ;
- La mise en place de Vaults ;
- L'utilisation de covenants ;
- Ou encore, l'utilisation de contrat de non équivocation.

OP_CHECKHASHVERIFY (CHV)

Nouvel opcode proposé en 2012 dans le BIP17 par Luke Dashjr qui permet de disposer des mêmes fonctionnalités que OP_EVAL ou P2SH. Il aurait dû permettre de hacher la fin du scriptSig, de comparer le résultat avec le haut de la pile et rendre la transaction invalide si les deux hachages ne correspondaient pas. Cet opcode n'a jamais été implémenté.

OP_CHECKLOCKTIMEVERIFY (0XB1)

Rend la transaction invalide sauf si toutes ces conditions sont réunies :

- La pile n'est pas vide ;
- La valeur du haut de la pile est supérieure ou égale à 0 ;
- Le type de timelock est le même entre le champ nLockTime et la valeur du haut de la pile (temps réel ou numéro de bloc) ;
- Le champ nSequence de l'input n'est pas égal à 0xffffffff ;
- La valeur du haut de la pile est supérieure ou égale à la valeur du champ nLockTime de la transaction.

Si une seule de ces conditions n'est pas remplie, le script contenant l'OP_CHECKLOCKTIMEVERIFY ne peut être satisfait. Si toutes ces conditions sont valides, alors OP_CHECKLOCKTIMEVERIFY agit comme un OP_NOP, c'est-à-dire qu'il ne fait aucune action sur le script. C'est un peu comme s'il disparaissait. OP_CHECKLOCKTIMEVERIFY impose donc une contrainte de temps sur la dépense de l'UTXO sécurisé avec le script le contenant. Il peut le faire soit sous la forme d'une date exprimée en temps Unix, soit sous la forme d'un numéro de bloc. Pour ce faire, il restreint les valeurs possibles pour le champs nLockTime de la transaction qui le dépense, et ce champs nLockTime restreint lui-même le moment où la transaction peut être incluse dans un bloc.

> *Cet opcode est un remplaçant d'OP_NOP. Il a été placé sur l'OP_NOP2. Il est souvent appelé par con acronyme « CLTV ». Attention, OP_CLTV ne doit pas être confondu avec le champs*

nLockTime d'une transaction. Le premier utilise le second, mais leurs natures et leurs actions sont différentes.

OP_CHECKMULTISIG (0XAE)

Vérifie plusieurs signatures contre plusieurs clés publiques. Il prend en entrée une série de N clés publiques et M signatures, où M peut être inférieur ou égal à N. OP_CHECKMULTISIG vérifie si au moins M signatures correspondent à M des N clés publiques. À noter qu'en raison d'un bug off-by-one historique, un élément supplémentaire est supprimé par OP_CHECKMULTISIG sur la pile. Cet élément est appelé « *dummy element* ». Pour éviter une erreur dans le ScriptSig, on inclue donc un OP_0 qui est un élément inutile afin de satisfaire la suppression et outrepasser le bug. Depuis le BIP147 (introduit avec SegWit en 2017), l'élément inutile consommé à cause du bug doit forcément être 0 pour que le script soit valide, car c'était un vecteur de malléabilité. Cet opcode a été supprimé dans Tapscript.

OP_CHECKMULTISIGVERIFY (0XAF)

Combine un OP_CHECKMULTISIG et un OP_VERIFY. Il prend plusieurs signatures et clés publiques pour vérifier que M parmi N signatures sont valides, comme le fait OP_CHECKMULTISIG. Puis, à l'instar d'OP_VERIFY, si la vérification échoue, le script s'arrête immédiatement avec une erreur. Si la vérification est réussie, le script continue sans pousser de valeur sur la pile. Cet opcode a été supprimé dans Tapscript.

OP_CHECKSEQUENCEVERIFY (0XB2)

Rend la transaction invalide si une seule de ces caractéristiques est observée :

- La pile est vide ;
- La valeur du haut de la pile est inférieure à 0 ;
- L'indicateur de désactivation de la valeur en haut de la pile est non défini et ;

- La version de la transaction est inférieure à 2 ou ; - L'indicateur de désactivation du champ nSequence de l'input est défini ou ; *- Le type de timelock n'est pas le même entre le champ nSequence et la valeur du haut de la pile (temps réel ou nombre de blocs) ;* - La valeur en haut de la pile est supérieure à la valeur du champ nSequence de l'input.

Si une ou plusieurs de ces caractéristiques est observée, le script contenant l'OP_CHECKSEQUENCEVERIFY ne peut être satisfait. Si toutes les conditions sont valides, alors OP_CHECKSEQUENCEVERIFY agit comme un OP_NOP, c'est-à-dire qu'il ne fait aucune action sur le script. C'est un peu comme s'il disparaissait. OP_CHECKSEQUENCEVERIFY impose donc une contrainte de temps relative sur la dépense de l'UTXO sécurisé avec le script le contenant. Il peut le faire soit sous la forme d'un temps réel, soit sous la forme d'un nombre de blocs. Pour ce faire, il restreint les valeurs possibles pour le champs nSequence de l'input qui le dépense, et ce champs nSequence restreint lui-même le moment où la transaction qui comprend cet input peut être incluse dans un bloc.

Cet opcode est un remplaçant d'OP_NOP. Il a été placé sur l'OP_NOP3. Il est souvent appelé par con acronyme « CSV ». Attention, OP_CSV ne doit pas être confondu avec le champs nSequence d'une transaction. Le premier utilise le second, mais leurs natures et leurs actions sont différentes.

OP_CHECKSIG (0XAC)

Vérifie la validité d'une signature par rapport à une clé publique donnée. Il prend les deux éléments du sommet de la pile : la signature et la clé publique, et évalue si la signature est correcte pour le hachage de la transaction et la clé publique spécifiée. Si la vérification est réussie, il pousse la valeur 1 (vrai) sur la pile, sinon 0 (faux). Cet opcode a été modifié dans Tapscript afin de vérifier des signatures de Schnorr.

OP_CHECKSIGADD (0XBA)

Extrait les trois valeurs en haut de la pile : une clé publique, un CScriptNum n et une signature. Si la signature n'est pas le vecteur vide et n'est pas valide, le script se termine avec une erreur. Si la signature est valide ou est le vecteur vide (OP_0), deux cas de figure se présente :

- Si la signature est le vecteur vide : un CScriptNum avec la valeur de n est poussé sur la pile et l'exécution continue ;
- Si la signature n'est pas le vecteur vide et demeure valide : un CScriptNum avec la valeur de n + 1 est poussé sur la pile et l'exécution continue.

Pour vulgariser, OP_CHECKSIGADD effectue une opération similaire à OP_CHECKSIG, mais au lieu de pousser vrai ou faux sur la pile, il ajoute 1 à la deuxième valeur en haut de la pile si la signature est valide, ou laisse cette valeur inchangée si la signature représente le vecteur vide.OP_CHECKSIGADD permet de créer les mêmes politiques multisignatures dans Tapscript qu'avec OP_CHECKMULTISIG et OP_CHECKMULTISIGVERIFY mais de manière vérifiable par lots, c'est-à-dire qu'il supprime le processus de recherche dans la vérification d'un multisig traditionnel et accélère donc la vérification tout en réduisant la charge opérationnelle sur les CPU des nœuds. Cet opcode a été ajouté dans Tapscript uniquement pour les besoins de Taproot.

OP_CHECKSIGVERIFY (0XAD)

Effectue la même opération que OP_CHECKSIG, mais si la vérification de la signature échoue, le script s'arrête immédiatement avec une erreur et la transaction est donc invalide. Si la vérification réussit, le script continue sans pousser de valeur 1 (vrai) sur la pile. Pour résumer, OP_CHECKSIGVERIFY réalise l'opération OP_CHECKSIG suivie de OP_VERIFY. Cet opcode a été modifié dans Tapscript afin de vérifier des signatures de Schnorr.

OP_CODESEPARATOR (0XAB)

Modifie le script de sortie en cours, en indiquant que seules les opérations qui suivent cet opcode seront prises en compte dans la vérification des signatures des entrées correspondantes. Cela permet de segmenter un script complexe en plusieurs parties, où chaque segment peut être signé indépendamment.

OP_DEPTH (0X74)

Pousse le nombre d'éléments dans la pile sur la pile elle-même. Si la pile contient N éléments, OP_DEPTH ajoutera le nombre N en tant que nouvel élément en haut de la pile.

OP_DROP (0X75)

Supprime l'élément situé au sommet de la pile. OP_DROP permet d'enlever des données devenues inutiles au cours de l'exécution du script.

OP_DUP (0X76)

Duplique le sommet de la pile. L'élément en haut de la pile est donc copié et la copie est placée en haut de la pile.

OP_ELSE (0X67)

Modifie le flux d'exécution dans un script conditionnel : il indique que les opérations qui le suivent doivent être exécutées si la condition précédente spécifiée par un OP_IF, un OP_NOTIF ou un autre OP_ELSE n'est pas remplie.

*Pour plus d'informations, voir la définition de **OP_IF**.*

OP_ENDIF (0X68)

Marque la fin d'une structure de contrôle conditionnelle initiée par un OP_IF ou un OP_NOTIF, normalement suivis par un ou plusieurs OP_ELSE. Il indique que l'exécution du script doit continuer au-delà de la structure conditionnelle, quelle que soit la branche qui a été exécutée. Autrement dit, OP_ENDIF permet de délimiter la fin des blocs conditionnels dans les scripts.

*Pour plus d'informations, voir la définition de **OP_IF**.*

OP_EQUAL (0X87)

Compare les deux valeurs les plus hautes de la pile et pousse 1 sur la pile si elles sont égales, sinon pousse 0. OP_EQUAL permet de vérifier l'égalité des données dans les scripts de transaction.

OP_EQUALVERIFY (0X88)

Combine les fonctions de OP_EQUAL et OP_VERIFY. Il vérifie d'abord l'égalité des deux valeurs supérieures de la pile, puis exige que le résultat soit non nul, faute de quoi la transaction est invalide. OP_EQUALVERIFY est notamment utilisé dans les scripts de vérification d'adresse.

OP_EVAL

Opcode proposé par Gavin Andresen en 2011. Il prend le script situé au sommet de la pile, l'exécute comme s'il faisait partie du scriptPubKey, et place son résultat sur la pile. OP_EVAL a été abandonné en raison de préoccupations liées à la complexité de cet opcode, qui sera finalement supplanté par P2SH.

OP_FALSE (0X00)

Identique à OP_0.

*Pour plus d'informations, voir la définition de **OP_0**.*

OP_FROMALTSTACK (0X6C)

Retire le sommet de la pile alternative (alt stack) et le place sur le sommet de la pile principale (main stack). Cet opcode est utilisé pour récupérer des données stockées temporairement sur la pile alternative. Pour simplifier, c'est l'opération inverse de OP_TOALTSTACK.

OP_GREATERTHAN (0XA0)

Compare les deux éléments au sommet de la pile et vérifie si le premier élément est supérieur au deuxième. Si le premier élément est supérieur au deuxième, il pousse 1 (vrai) sur la pile, sinon, il pousse 0 (faux).

OP_GREATERTHANOREQUAL (0XA2)

Compare les deux éléments au sommet de la pile et vérifie si le premier élément est supérieur ou égal au deuxième. Si le premier élément est supérieur ou égal au deuxième, il pousse 1 (vrai) sur la pile, sinon, il pousse 0 (faux).

OP_HASH160 (0XA9)

Prend l'élément en haut de la pile et le remplace par son hachage en utilisant simultanément les fonctions SHA256 et RIPEMD160. Ce processus en deux étapes génère une empreinte de 160 bits.

*Pour plus d'informations, voir la définition de **SHA256** et **RIPEMD160**.*

OP_HASH256 (0XAA)

Prend l'élément en haut de la pile et le remplace par son hachage en utilisant deux fois la fonction SHA256. L'entrée est hachée une première fois avec SHA256 et le condensat est haché une seconde fois avec SHA256. Ce processus en deux étapes génère une empreinte de 256 bits.

*Pour plus d'informations, voir la définition de **SHA256**.*

OP_IF (0X63)

Exécute la portion suivante du script si la valeur au sommet de la pile est non nulle (vraie). Si la valeur est nulle (fausse), ces opérations sont sautées, passant directement à celles après OP_ELSE, s'il est présent. OP_IF permet d'initier une structure de contrôle conditionnelle dans un script. Il détermine le flux de contrôle dans un script en fonction d'une condition fournie au moment de l'exécution de la transaction. OP_IF s'utilise avec OP_ELSE et OP_ENDIF de la manière suivante : `<condition> OP_IF <opérations si la condition est vraie> OP_ELSE <opérations si la condition est fausse> OP_ENDIF`.

OP_IFDUP (0x73)

Duplique le sommet de la pile si celui-ci est non nul. Si la valeur en haut de la pile est vraie (c'est-à-dire non nulle), cette valeur est dupliquée sur la pile ; sinon, la pile reste inchangée.

OP_LESSTHAN (0X9F)

Compare les deux éléments au sommet de la pile et vérifie si le premier élément est inférieur au deuxième. Si le premier élément est inférieur au deuxième, il pousse 1 (vrai) sur la pile, sinon, il pousse 0 (faux).

OP_LESSTHANOREQUAL (0XA1)

Compare les deux éléments au sommet de la pile et vérifie si le premier élément est inférieur ou égal au deuxième. Si le premier élément est inférieur ou égal au deuxième, il pousse 1 (vrai) sur la pile, sinon, il pousse 0 (faux).

OP_MAX (0XA4)

Sélectionne le plus grand des deux éléments en haut de la pile et le pousse sur la pile. Cette opération conserve uniquement la plus grande des deux valeurs au sommet.

OP_MIN (0XA3)

Sélectionne le plus petit des deux éléments en haut de la pile et le pousse sur la pile. Cette opération conserve uniquement la plus petite des deux valeurs au sommet.

OP_NEGATE (0X8F)

Inverse le signe de l'élément supérieur de la pile. Si la valeur est positive, elle devient négative, et vice versa.

OP_NIP (0x77)

Supprime l'élément situé juste en dessous du sommet de la pile (le second en partant du haut).

OP_NOP (0X61)

Ne produit aucun effet sur la pile ou l'état du script. Il ne fait aucun déplacement, aucune vérification, ni aucune modification. Il ne fait juste rien, à moins que l'on ait décidé qu'il fasse quelque chose via un soft fork. En effet, depuis leurs modifications par Satoshi Nakamoto en 2010, les commandes OP_NOP (OP_NOP1 (0XB0) jusqu'à OP_NOP10 (0XB9)) sont utilisées pour ajouter de nouveaux opcodes sous forme de soft fork. Ainsi, l'OP_NOP2 (0XB1) et l'OP_NOP3 (0XB2) ont déjà été utilisés pour implémenter respectivement l'OP_CHECKLOCKTIMEVERIFY et l'OP_CHECKSEQUENCEVERIFY. On les utilise en combinaison avec OP_DROP afin de supprimer de la pile les valeurs temporelles associées, et ainsi pouvoir continuer l'exécution du script, que le nœud soit à jour ou non. Les OP_NOP permettent donc d'insérer un point d'interruption dans un script pour vérifier des conditions supplémentaires déjà existantes ou pouvant être ajoutées avec de futurs soft fork. Depuis Tapscript, l'utilisation des OP_NOP a été remplacée par l'utilisation des OP_SUCCESS étant plus efficace.

*Pour plus d'informations, voir la définition de **OP_SUCCESS**.*

OP_NOT (0X91)

Inverse la valeur booléenne du sommet de la pile : si cette valeur est non nulle, l'opérateur la remplace par 0, sinon par 1.

OP_NOTIF (0X64)

Fonctionne de manière opposée à OP_IF, exécutant la portion suivante du script si la valeur au sommet de la pile est nulle (fausse).

*Pour plus d'informations, voir la définition de **OP_IF**.*

OP_NUMEQUAL (0X9C)

Compare les deux éléments au sommet de la pile pour vérifier s'ils sont numériquement égaux. Si les valeurs sont égales, il pousse 1 (vrai) sur la pile, sinon, il pousse 0 (faux).

OP_NUMEQUALVERIFY (0X9D)

Combine les opérations OP_NUMEQUAL et OP_VERIFY. Il compare numériquement les deux éléments au sommet de la pile. Si les valeurs sont égales, OP_NUMEQUALVERIFY supprime le résultat vrai de la pile et continue l'exécution du script. Si les valeurs ne sont pas égales, le résultat est faux et le script échoue immédiatement.

OP_NUMNOTEQUAL (0X9E)

Compare les deux éléments au sommet de la pile pour vérifier s'ils sont numériquement non égaux. Si les valeurs ne sont pas égales, il pousse 1 (vrai) sur la pile, sinon, il pousse 0 (faux). C'est l'inverse de OP_NUMEQUAL.

OP_OVER (0X78)

Duplique le deuxième élément à partir du haut de la pile et le place sur le haut de la pile.

OP_PICK (0X79)

Duplique un élément de la pile et le place en haut de la pile, en fonction de la profondeur spécifiée par la valeur en haut de la pile avant l'opération. Par exemple, si la valeur en haut de la pile est 4, OP_PICK va dupliquer le quatrième élément de la pile en partant du sommet, et il va placer cette copie au sommet.

OP_PUSHDATA1 (0x4C)

Pousse une certaine quantité de données sur la pile. Il est suivi d'un octet qui indique la longueur des données à pousser (jusqu'à 255 octets). Cet opcode est utilisé pour inclure des données de taille variable dans les scripts.

OP_PUSHDATA2 (0x4D)

Permet de pousser une grande quantité de données sur la pile. Il est suivi de deux octets (petit-boutistes) qui spécifient la longueur des données à pousser (jusqu'à 65535 octets). Il est utilisé pour insérer des données plus volumineuses dans les scripts.

OP_PUSHDATA4 (0x4E)

Permet de pousser une très grande quantité de données sur la pile. Il est suivi de quatre octets (petit-boutistes) qui indiquent la longueur des données à pousser (jusqu'à 4 294 967 295 octets). Cet opcode est utilisé pour insérer de très grandes données dans les scripts, bien que son usage soit extrêmement rare en raison des limitations pratiques de la taille des transactions.

OP_RETURN (0X6A)

Signale un script invalide, ce qui rend l'output qui le contient comme non dépensable de manière prouvée. Les nœuds du réseau peuvent donc supprimer cet output de leurs UTXO set. OP_RETURN est souvent utilisé pour inscrire des données arbitraires dans la blockchain.

OP_RIPEMD160 (0XA6)

Prend l'élément en haut de la pile et le remplace par son hachage en utilisant la fonction RIPEMD160.

*Pour plus d'informations, voir la définition de **RIPEMD160**.*

OP_ROLL (0X7A)

Déplace un élément de la pile et en haut de la pile, en fonction de la profondeur spécifiée par la valeur en haut de la pile avant l'opération. Par exemple, si la valeur en haut de la pile est 4, OP_ROLL va sélectionner le quatrième élément de la pile en partant du sommet, et il va déplacer cette valeur au sommet. OP_ROLL joue le même rôle que OP_PICK, mis à part qu'il retire l'élément de sa position initiale.

OP_ROT (0X7B)

Déplace au sommet de la pile le troisième élément à partir du sommet de la pile. Les deux éléments qui étaient au-dessus de lui sont poussés en dessous dans l'ordre inverse.

OP_SHA1 (0XA7)

Prend l'élément en haut de la pile et le remplace par son hachage en utilisant la fonction SHA1. L'utilisation de cette fonction est aujourd'hui déconseillée dans un cadre sécurisé.

OP_SHA256 (0XA8)

Prend l'élément en haut de la pile et le remplace par son hachage en utilisant la fonction SHA256.

*Pour plus d'informations, voir la définition de **SHA256**.*

OP_SIZE (0X82)

Mesure la taille en nombre d'octets de l'élément en haut de la pile et renvoie cette taille au sommet de la pile, sans pour autant modifier l'élément analysé en lui-même.

OP_SUB (0X94)

Soustrait les deux éléments au sommet de la pile. Il prend les deux valeurs en haut de la pile, il les soustrait et il les remplace par le résultat.

OP_SUCCESS

Les OP_SUCCESS représentent une série d'opcodes qui ont été désactivés par le passé et qui sont dorénavant réservés pour une utilisation future dans Tapscript. Leur but final est de faciliter des mises à jour et des extensions du langage script, en permettant l'introduction de nouvelles fonctionnalités via

des soft forks. Lorsqu'un de ces opcodes est rencontré dans un script, il indique un succès automatique de cette partie du script, peu importe les données ou les conditions présentes. Cela signifie que le script continue son exécution sans échec, indépendamment des opérations précédentes. Ainsi, lorsque l'on ajoute un nouvel opcode sur un `OP_SUCCESS`, cela représente forcément l'ajout d'une règle plus restrictive que la règle précédente. Les nœuds à jour peuvent alors vérifier le respect de cette règle et les nœuds pas à jour ne refuseront pas les transactions associées et les blocs qui les incluent, car l'`OP_SUCCESS` valide cette partie du script. Il n'y a donc pas de hard fork. En comparaison, les `OP_NOP` (*No Operation*) servent également de marqueurs de place dans le script, mais ils n'ont aucun effet sur l'exécution du script. Lorsqu'un `OP_NOP` est rencontré, le script continue simplement sans modifier l'état de la pile ou le déroulement du script. La différence clé est donc dans leur impact sur l'exécution : `OP_SUCCESS` garantit un passage réussi à travers cette portion du script, tandis que `OP_NOP` est neutre, n'affectant ni la pile ni le flux du script. Ces opcodes sont désignés par `OP_SUCCESSN` où `N` est un numéro permettant de différencier les `OP_SUCCESS`.

OP_SWAP (0X7C)

Échange les deux éléments en haut de la pile. L'élément qui était au sommet est déplacé en deuxième position, et l'élément qui était en deuxième position est placé au sommet de la pile.

OP_TOALTSTACK (0X6B)

Prend le sommet de la pile principale (main stack) et le déplace vers la pile alternative (alt stack). Cet opcode permet de stocker temporairement des données à part pour une utilisation ultérieure dans le script. L'élément déplacé est donc supprimé de la pile principale. On utilisera ensuite `OP_FROMALTSTACK` pour le remettre au sommet de la pile principale.

OP_TRUE (0X51)

Identique à OP_1.

*Pour plus d'informations, voir la définition de **OP_1**.*

OP_TUCK (0X7D)

Copie l'élément situé au sommet de la pile et l'insère entre le deuxième élément et le troisième élément de la pile. Par exemple, si la pile est `D C B A`, `OP_TUCK` va dupliquer le sommet `A` et le placer en troisième position. La pile en sortie sera : `D C A B A`.

OP_VER (0X62)

Permettait de pousser la version du client sur la pile. Cet opcode a été désactivé car s'il avait été utilisé, chaque mise à jour aurait conduit à un hard fork. Le BIP342 a modifié cet opcode en `OP_SUCCESS`.

OP_VERIFY (0X69)

Exige que la valeur du sommet de la pile soit non nulle (vraie). La transaction est invalide si ce n'est pas le cas. `OP_VERIFY` est utilisé pour confirmer les conditions des scripts.

OP_WITHIN (`0XA5`)

Vérifie si le premier élément en haut de la pile se trouve dans l'intervalle défini par les deuxième et troisième éléments supérieurs. Autrement dit, `OP_WITHIN` vérifie si le premier élément est supérieur ou égal au deuxième et inférieur au troisième. Si cette condition est vraie, il pousse `1` (`vrai`) sur la pile, sinon, il pousse `0` (`faux`).

OPCODES

Ensemble des commandes utilisées dans le système Script de Bitcoin. Script est un langage de programmation à pile utilisé pour établir des conditions de dépense, et donc, indirectement, sécuriser des bitcoins. Les instructions utilisées en langage Script sont appelées « OPcodes ». Ce sont des opérateurs logiques et des commandes pour manipuler la pile (stack). Ces instructions spécifiques sont exécutées par les nœuds du réseau lors de l'ajout d'une transaction à la blockchain. Script est un langage non-Turing complet. Il peut-être catégorisé comme un langage de niveau intermédiaire (presque bas niveau) inspiré du Forth.

> *« OPcode » peut être traduit en français par « code opératoire ». Dans la pratique, on utilise directement le terme « OPcode » dans le langage courant.*

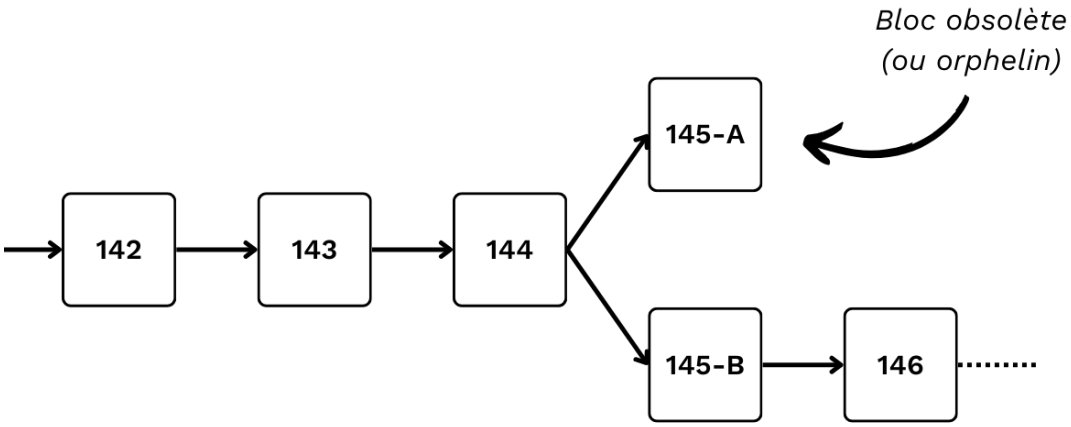
ORACLE

Source d'informations tierce qui fournit des données du monde réel pouvant être interprétées sur Bitcoin. Les oracles permettent aux contrats intelligents, tels que les DLC, d'exécuter des conditions contractuelles en fonction d'événements extérieurs. En général, ils fournissent une signature spécifique qui correspond au résultat d'un événement. Cette signature est ensuite utilisée pour compléter et rendre valide une transaction d'exécution qui envoie les bitcoins à la partie qui est censée les recevoir selon les conditions du contrat intelligent.

> *Pour plus d'informations, voir la définition de **DLC (DISCREET LOG CONTRACT)**.*

ORPHELIN (BLOC)

Théoriquement, un bloc orphelin désigne un bloc valide réceptionné par un nœud qui n'a pas encore acquis le bloc parent, c'est-à-dire le précédent dans la chaîne. Ce bloc, bien que valide, demeure isolé localement en tant qu'orphelin. Cependant, dans l'usage courant, l'expression « bloc orphelin » fait souvent référence à un bloc sans enfant : un bloc valide mais non retenu dans la chaîne principale de Bitcoin. Il se produit lorsque deux mineurs trouvent un bloc valide sur une même hauteur de chaîne durant un court laps de temps et le diffusent sur le réseau. Le réseau finit par choisir un seul bloc à inclure dans la chaîne, selon le principe de la chaîne avec le plus de quantité de travail accumulé, rendant l'autre « orphelin ».



Personnellement, je préfère employer le terme de « bloc orphelin » pour parler d'un bloc sans parent et le terme de « bloc obsolète » (stale block) pour désigner un bloc qui n'a pas d'enfant. Je trouve cela plus logique et compréhensible, bien qu'une majorité de bitcoiners ne suivent pas cet usage. Pour plus d'informations, voir la définition de ***OBSOLÈTE (BLOC)****.*

OU EXCLUSIF

Traduction française du terme « *exclusive or* » (XOR).

Pour plus d'informations, voir la définition du terme ***XOR****.***

OUTBOUND CAPACITY

Désigne la quantité maximale de bitcoins qu'un nœud peut envoyer à travers un canal spécifique sur le Lightning Network. Elle dépend des fonds que le nœud a engagés dans le canal lors de son ouverture, ou qu'il a reçus lors d'un paiement Lightning.

En français, on peut le traduire par « capacité sortante ».

OUTPUT

Traduction anglaise de « sortie ». Dans le contexte de Bitcoin, une « sortie » au sein d'une transaction fait référence aux *Unspent Transaction Outputs* (UTXO) qui sont créés comme fonds de destination pour le paiement. Plus précisément, il s'agit d'un mécanisme par lequel une transaction distribue des fonds. Une transaction prend des UTXO, c'est-à-dire des morceaux de bitcoins, comme « inputs » (entrées) et crée de nouveaux UTXO comme « outputs » (sorties). Ces outputs stipulent une certaine quantité de bitcoins, souvent attribués à une adresse spécifique, ainsi que les conditions sous lesquelles ces fonds peuvent être dépensés ultérieurement. Le rôle de la transaction Bitcoin est donc de consommer des UTXO en entrées, et de créer des nouveaux UTXO en sorties. La différence entre les deux correspond aux frais de transactions qui peuvent être récupérés par le

mineur gagnant du bloc. Un UTXO est, en essence, la sortie d'une transaction précédente qui n'a pas encore été dépensée. Les outputs de transaction sont donc les créations de nouveaux UTXO qui seront, à leur tour, potentiellement utilisés comme inputs dans les transactions futures. D'un point de vue plus large, en informatique, le terme « output » ou « sortie » désigne généralement les données en résultat d'une fonction, d'un algorithme, ou d'un système. Par exemple, lorsque l'on passe une donnée dans une fonction de hachage cryptographique, cette information est nommée « entrée » ou « input », et le résultat est nommé « sortie » ou « output ».

OUTPUT LINKING

Synonyme parfois utilisé pour parler de réutilisation d'adresse. L'output linking se réfère à la pratique d'utiliser une même adresse de réception pour bloquer plusieurs UTXO, parfois au sein de plusieurs transactions différentes. Les bitcoins sont généralement bloqués à l'aide d'une paire de clés cryptographique qui correspond à une adresse unique. Puisque la blockchain est publique, il est facile de pouvoir consulter quelles adresses sont associées à combien de bitcoins. En cas de réutilisation d'une même adresse pour plusieurs paiements, on peut raisonnablement imaginer que tous les UTXO associés appartiennent à une même entité. La réutilisation d'adresse pause donc un problème pour la vie privée de l'utilisateur. Elle permet de faire des liens déterministes entre plusieurs transactions et plusieurs UTXO, ainsi que de perpétuer un traçage de fonds on-chain.

Pour plus d'informations, voir la définition de RÉUTILISATION D'ADRESSE.

P

P2PK

P2PK est le sigle pour *Pay to Public Key* (en français « payer à une clé publique »). C'est un modèle de script standard utilisé sur Bitcoin pour établir des conditions de dépenses sur un UTXO. Il permet de bloquer des bitcoins directement sur une clé publique, plutôt que sur une adresse. Techniquement, le script P2PK contient une clé publique et une instruction qui exige une signature numérique correspondante pour débloquer les fonds. Lorsque le propriétaire souhaite dépenser les bitcoins, il doit fournir une signature produite avec la clé privée associée. Cette signature est vérifiée avec l'algorithme ECDSA (Elliptic Curve Digital Signature Algorithm). P2PK était souvent utilisé dans les premières versions de Bitcoin, notamment par Satoshi Nakamoto. Il n'est presque plus utilisé à ce jour.

P2PKH

P2PKH est le sigle pour *Pay to Public Key Hash* (en français « payer au hachage d'une clé publique »). C'est un modèle de script standard utilisé pour établir des conditions de dépenses sur un UTXO. Il permet de bloquer des bitcoins sur un hachage d'une clé publique, c'est-à-dire sur une adresse de réception. Ce script est associé au standard Legacy, et a été introduit dès les premières versions de Bitcoin par Satoshi Nakamoto. À la différence du P2PK, où la clé publique est explicitement incluse dans le script, le P2PKH fait appel à une empreinte cryptographique de la clé publique, avec quelques métadonnées, également nommée « adresse de réception ». Ce script inclut le hachage RIPEMD160 du SHA256 de la clé publique et stipule que, pour accéder aux fonds, le destinataire doit fournir une clé publique correspondant à ce hachage, ainsi qu'une signature numérique valide générée à partir de la clé privée associée. Les adresses P2PKH sont encodées en utilisant le format Base58Check, ce qui leur confère une robustesse contre les erreurs typographiques grâce à l'utilisation d'une somme de contrôle. Ces adresses débutent systématiquement par le chiffre 1.

P2P TRANSPORT V2

Nouvelle version du protocole de transport Bitcoin P2P intégrant le chiffrement opportuniste pour améliorer la confidentialité et la sécurité des communications entre les nœuds. Cette amélioration vise à résoudre plusieurs problématiques de la version de base du protocole P2P, notamment en rendant les données échangées indiscernables pour un observateur passif (tel qu'un fournisseur d'accès à internet), réduisant ainsi les risques de censure et d'attaques par détection de motifs spécifiques dans les paquets de données. Le chiffrement mis en place n'inclut pas d'authentification afin de ne pas ajouter de complexité inutile, et de ne pas compromettre le fait que la connexion au réseau reste sans permission. Ce nouveau protocole de transport P2P offre néanmoins une meilleure sécurité contre les attaques passives et rend les attaques actives nettement plus coûteuses et détectables (notamment les attaques MITM). L'introduction d'un flux de données pseudo-aléatoire complique la tâche des attaquants souhaitant censurer ou manipuler les communications. Le transport P2P V2 a été inclus en option (désactivé par défaut) dans la version 26.0 de Bitcoin Core, déployée en décembre 2023. Il peut être activé avec l'option `v2transport=1` dans le fichier de configuration.

P2MS

P2MS est le sigle pour *Pay to Multisig* (en français « payer aux multiples signatures »). C'est un modèle de script standard utilisé pour établir des conditions de dépenses sur un UTXO. Il permet de bloquer des bitcoins à l'aide de plusieurs clés publiques. Pour dépenser ces bitcoins, il faut réaliser une signature avec un nombre prédéfini de clés privées associées. Par exemple, un P2MS 2/3 dispose de 3 clés publiques avec 3 clés privées secrètes associées. Pour dépenser les bitcoins bloqués avec ce script P2MS, il faut réaliser une signature avec au moins 2 parmi les 3 clés privées. C'est un

système de sécurisation à seuil (threshold). Ce script a été inventé en 2011 par Gavin Andresen alors qu'il venait de récupérer la maintenance du client principal de Bitcoin. Aujourd'hui, le P2MS n'est utilisé qu'à la marge par certaines applications. L'extrême majorité des multisignatures modernes emploient d'autres scripts comme le P2SH ou le P2WSH. Par rapport à ceux-ci, le P2MS est extrêmement trivial. Les clés publiques le constituant sont dévoilées dès la réception de la transaction. L'utilisation d'un P2MS est également plus chère que les autres scripts multisignature.

> *Les P2MS sont souvent nommés « bare-multisig », ce qui peut être traduit en français par « multi-signature nu », ou « multi-signature brut ». Au début de l'année 2023, les scripts P2MS étaient au centre d'une polémique à cause de leur utilisation détournée au sein du protocole Stamps. Leur impact sur l'UTXO SET était notamment pointé du doigt.*

P2SH

P2SH est le sigle pour *Pay to Script Hash* (en français « payer au hachage du script »). C'est un modèle de script standard utilisé pour établir des conditions de dépenses sur un UTXO. Contrairement aux scripts P2PK et P2PKH, où les conditions de dépense sont prédéfinies, P2SH permet l'intégration de conditions de dépense arbitraires et de fonctionnalités additionnelles au sein d'un script de transaction. Techniquement, dans une transaction P2SH, le ScriptPubKey contient l'empreinte cryptographique d'un script de rachat, plutôt que de conditions de dépense explicites. Cette empreinte est obtenue en utilisant un hachage SHA256. Lors de l'envoi de bitcoins à une adresse P2SH, le script de rachat sous-jacent n'est pas révélé. Seule son empreinte est incluse dans la transaction. Les adresses P2SH sont encodées en Base58Check et commencent par le chiffre 3. Lorsque le destinataire souhaite dépenser les bitcoins reçus, il doit fournir un script de rachat correspondant à l'empreinte, ainsi que les données nécessaires pour satisfaire les conditions de ce script. Par exemple, dans un P2SH multisignatures, le script pourrait exiger des signatures de plusieurs clés privées. L'utilisation de P2SH confère une flexibilité considérable, car il permet la construction de scripts arbitraires sans que l'expéditeur ait à en connaître les détails. P2SH est introduit en 2012 avec le BIP16.

P2SH-P2WPKH

- P2SH-P2WPKH est le sigle pour *Pay to Script Hash - Pay to Witness Public Key Hash* (en français « payer au hachage du script - payer au témoin du hachage de la clé publique »). C'est un modèle de script standard utilisé pour établir des conditions de dépenses sur un UTXO, également connu sous le nom de « Nested SegWit ». P2SH-P2WPKH a été introduit avec l'implémentation de SegWit en août 2017. Ce script décrit un P2WPKH enveloppé au sein d'un P2SH . Il verrouille des bitcoins sur la base du hachage d'une clé publique. La différence avec P2WPKH simple est que le script est enveloppé dans le redeemScript d'un P2SH classique. Ce script a été créé au lancement de SegWit pour faciliter son adoption. Il permet d'utiliser ce nouveau standard, même sur des wallets pas encore compatibles nativement avec SegWit. C'est une sorte de script de transition vers la nouvelle norme. Aujourd'hui, il n'est donc plus très pertinent d'utiliser ce type de scripts SegWit wrappés, puisque la plupart des wallets ont implémenté du SegWit natif. Les adresses P2SH-P2WPKH sont écrites en utilisant l'encodage Base58Check et commencent toujours par 3 , comme n'importe quelle adresse P2SH .

  « P2SH-P2WPKH » est également parfois appelé « P2WPKH-nested-in-P2SH ».

P2SH-P2WSH

- P2SH-P2WSH est le sigle pour *Pay to Script Hash - Pay to Witness Script Hash* (en français « payer au hachage du script - payer au témoin du hachage du script »). C'est un modèle de script standard utilisé pour établir des conditions de dépenses sur un UTXO, également

connu sous le nom de « Nested SegWit ». P2SH-P2WSH a été introduit avec l'implémentation de SegWit en août 2017. Ce script décrit un P2WSH enveloppé au sein d'un P2SH . Il verrouille des bitcoins sur la base du hachage d'un script. La différence avec P2WSH simple est que le script est enveloppé dans le redeemScript d'un P2SH classique. Ce script a été créé au lancement de SegWit pour faciliter son adoption. Il permet d'utiliser ce nouveau standard, même sur des wallets pas encore compatibles nativement avec SegWit. C'est une sorte de script de transition vers la nouvelle norme. Aujourd'hui, il n'est donc plus très pertinent d'utiliser ce type de scripts SegWit wrappés, puisque la plupart des wallets ont implémenté du SegWit natif. Les adresses P2SH-P2WSH sont écrites en utilisant l'encodage Base58Check et commencent toujours par 3 , comme n'importe quelle adresse P2SH .

P2TR

P2TR est le sigle pour *Pay to Taproot* (en français « payer à la racine »). C'est un modèle de script standard utilisé pour établir des conditions de dépenses sur un UTXO. P2TR a été introduit avec l'implémentation de Taproot en novembre 2021. P2TR utilise le protocole de Schnorr pour agréger des clés cryptographiques, ainsi que des arbres de Merkle pour des scripts alternatifs, connus sous le nom de MAST (*Merkelized Alternative Script Tree*). Contrairement aux transactions traditionnelles où les conditions de dépense sont exposées publiquement (parfois à la réception, parfois à la dépense) P2TR permet de masquer des scripts complexes derrière une seule clé publique apparente. Techniquement, un script P2TR verrouille des bitcoins sur une clé publique Schnorr unique, dénommée K. Cependant, cette clé K est en réalité un agrégat d'une clé publique P et d'une clé publique M, cette dernière étant calculée à partir de la racine de Merkle d'une liste de ScriptPubKeys. L'agrégation de clés est réalisée à l'aide du protocole de signature de Schnorr. Les bitcoins verrouillés avec un script P2TR peuvent être dépensés de deux manières distinctes : soit en publiant une signature pour la clé publique P, soit en satisfaisant l'un des scripts contenus dans l'arbre de Merkle. La première option est appelée « key path » (chemin de clé) et la seconde « script path » (chemin de script). Ainsi, P2TR permet aux utilisateurs d'envoyer des bitcoins soit à une clé publique, soit à plusieurs scripts de leur choix. Un autre avantage de ce script est que, bien qu'il y ait de multiples façons de dépenser une sortie P2TR, seule celle qui est utilisée doit être révélée à la dépense, permettant ainsi aux alternatives inutilisées de rester privées. Par exemple, grâce à l'agrégation des clés Schnorr, la clé publique P peut elle-même être une clé agrégée, représentant éventuellement un multisig. P2TR est une sortie SegWit de version 1, ce qui signifie que les signatures pour les entrées P2TR sont stockées dans le témoin d'une transaction, et non dans le ScriptSig. Les adresses P2TR utilisent un encodage Bech32m et commencent par bc1p.

P2WPKH

P2WPKH est le sigle pour *Pay to Witness Public Key Hash* (en français « payer au témoin du hachage de la clé publique »). C'est un modèle de script standard utilisé pour établir des conditions de dépenses sur un UTXO. P2WPKH a été introduit avec l'implémentation de SegWit en août 2017. Ce script est similaire à P2PKH (Pay to Public Key Hash), en ce sens qu'il verrouille également des bitcoins sur la base du hachage d'une clé publique, c'est-à-dire d'une adresse de réception. La différence réside dans la manière dont les signatures et les scripts sont inclus dans la transaction. Dans le cadre de P2WPKH, les informations du script de signature (ScriptSig) sont déplacées de la structure traditionnelle de la transaction vers une section distincte appelée Witness (témoin). Ce déplacement est une caractéristique de la mise à jour SegWit (*Segragated Witness*). Cette technique présente l'avantage de réduire la taille des données de transaction dans le corps principal, tout en conservant les informations de script nécessaires à la validation dans une section séparée. Par conséquent, les transactions P2WPKH sont généralement moins coûteuses en termes de frais par rapport aux transactions Legacy. Les adresses P2WPKH sont écrites en utilisant l'encodage Bech32, ce qui contribue à une écriture plus

concise et moins sujette aux erreurs typographiques grâce à la somme de contrôle sous forme de code BCH. Ces adresses commencent toujours par `bc1q`, ce qui permet de les distinguer facilement des adresses de réception Legacy. `P2WPKH` est une sortie SegWit de version 0.

P2WSH

`P2WSH` est le sigle pour *Pay to Witness Script Hash* (en français « payer au témoin du hachage du script »). C'est un modèle de script standard utilisé pour établir des conditions de dépenses sur un UTXO. `P2WSH` a été introduit avec l'implémentation de SegWit en août 2017. Ce script est similaire à `P2SH` (*Pay to Public Script Hash*), en ce sens qu'il verrouille également des bitcoins sur la base du hachage d'un script. La différence réside dans la manière dont les signatures et les scripts sont inclus dans la transaction. Pour dépenser les bitcoins sur ce type de script, le bénéficiaire doit fournir le script d'origine, appelé `RedeemScript`, ainsi que les signatures requises. Ce mécanisme permet d'implémenter des conditions de dépense plus sophistiquées, telles que des multisig. Dans le cadre de `P2WSH`, les informations du script de signature (`ScriptSig`) sont déplacées de la structure traditionnelle de la transaction vers une section distincte appelée `Witness` (témoin). Ce déplacement est une caractéristique de la mise à jour SegWit (*Segragated Witness*). Cette technique présente l'avantage de réduire la taille des données de transaction dans le corps principal, tout en conservant les informations de script nécessaires à la validation dans une section séparée. Par conséquent, les transactions `P2WSH` sont généralement moins coûteuses en termes de frais par rapport aux transactions `P2SH`. Les adresses `P2WSH` sont écrites en utilisant l'encodage `Bech32`, ce qui contribue à une écriture plus concise et moins sujette aux erreurs typographiques grâce à la somme de contrôle sous forme de code BCH. Ces adresses commencent toujours par `bc1q`, ce qui permet de les distinguer facilement des adresses de réception Legacy. `P2WSH` est une sortie SegWit de version 0.

PAIR-À-PAIR (P2P)

Fait référence à un modèle de communication et de distribution de données dans lequel les participants, souvent appelés nœuds ou pairs, partagent leurs ressources (comme des fichiers, de la puissance de traitement, de la bande passante, des actifs…) directement entre eux, sans nécessiter d'intermédiaire centralisé. Dans un système P2P, chaque participant agit simultanément comme client (consommateur de ressources) et serveur (fournisseur de ressources). Dans le contexte de Bitcoin, le terme pair-à-pair revêt une importance particulière. Le réseau Bitcoin fonctionne selon un modèle P2P, où les nœuds sont responsables de la validation des transactions et de la conservation de la blockchain. Cela signifie que, contrairement aux systèmes bancaires traditionnels qui dépendent d'entités centralisées, Bitcoin opère sur une structure distribuée où aucune entité unique ne détient le contrôle. Les nœuds du réseau Bitcoin communiquent entre eux pour diffuser les transactions et les blocs, et trouver un consensus sur l'état du registre.

PAIR ENTRANT

Nœud du réseau Bitcoin qui initie une connexion vers votre nœud sans intervention de votre part. Bitcoin Core autorise au maximum 125 pairs entrants par défaut, afin de faciliter la connectivité au sein du réseau. Les pairs entrants sont considérés avec prudence, car on ne peut pas être sûr qu'ils sont honnêtes, du fait qu'ils soient initiés par des tiers. Les pairs entrants et sortants partagent le même type d'informations. La principale différence entre les pairs entrants et sortants réside non pas dans le type d'informations échangées, mais dans la manière dont ces connexions sont établies.

> *La traduction anglaise de « pair entrant » est « inbound peer » ou « incoming connection ».*

PAIR SORTANT

Nœud vers lequel votre propre nœud Bitcoin établit activement une connexion. Par défaut, un nœud tente de se connecter à 8 pairs sortants. Ces connexions sont privilégiées et considérées comme plus fiables que les pairs entrant, car elles sont choisies par le nœud. Les pairs entrants et sortants partagent le même type d'informations. La principale différence entre les pairs entrants et sortants réside non pas dans le type d'informations échangées, mais dans la manière dont ces connexions sont établies.

> *La traduction anglaise de « pair sortant » est « outbound peer » ou « outgoing connection ».*

PASSPHRASE (BIP39)

Mot de passe optionnel qui, combiné à la phrase de récupération, offre une couche de sécurité supplémentaire pour les portefeuilles Bitcoin déterministes et hiérarchiques. Les portefeuilles HD sont généralement générés à partir d'une phrase de récupération constituée de 12 ou de 24 mots. Cette phrase de récupération est cruciale, car elle permet de restaurer l'ensemble des clés d'un portefeuille en cas de perte. Cependant, elle constitue un point de défaillance unique (SPOF). Si elle est compromise, les actifs sont en danger. C'est là qu'intervient la passphrase. C'est un mot de passe optionnel, choisi par l'utilisateur, qui s'ajoute à la phrase de récupération pour renforcer la sécurité du portefeuille. À ne pas confondre avec un code PIN ou un mot de passe ordinaire, la passphrase joue un rôle dans la dérivation des clés cryptographiques. Elle fonctionne en tandem avec la phrase de récupération, modifiant la graine à partir de laquelle sont générées les clés. Ainsi, même si une personne obtient votre phrase de récupération, sans la passphrase, elle ne peut pas accéder à vos fonds. L'utilisation d'une passphrase crée essentiellement un nouveau portefeuille avec des clés distinctes. Modifier (même légèrement) la passphrase générera un portefeuille différent. La passphrase est arbitraire et peut-être n'importe quelle combinaison de caractères choisie par l'utilisateur. L'utilisation d'une passphrase offre plusieurs avantages. Tout d'abord, elle réduit les risques liés à la compromission de la phrase de récupération en nécessitant un second facteur pour accéder aux fonds. Ensuite, elle peut être utilisée stratégiquement pour créer des portefeuilles d'appât contenant de petites quantités de bitcoins, dans le cas d'une contrainte physique pour voler vos bitcoins. Enfin, son utilisation est intéressante lorsque l'on souhaite maitriser le caractère aléatoire de la génération de la graine du portefeuille HD. La passphrase doit être suffisamment complexe pour résister aux attaques par brute force et doit être sauvegardée de manière fiable. La perte de la passphrase peut entraîner l'incapacité d'accéder aux fonds, tout comme la perte de la phrase de récupération.

> *La passphrase est parfois également nommée : « two-factor seed phrase », « password », « seed extension », « extention word » ou encore « 13ème ou 25ème mot ». Notons qu'il existe deux types de passphrases sur Bitcoin. La plus connue est celle décrite ci-dessus, qui dépend du BIP39, et qui permet de sécuriser tout un portefeuille HD entier. Toutefois, le BIP38 avait également spécifié une manière de sécuriser une clé privée unique à l'aide d'une passphrase. Ce second type de passphrase n'est presque plus utilisé aujourd'hui. Pour plus d'informations sur cette autre passphrase, voir la définition de* ***BIP38****.*

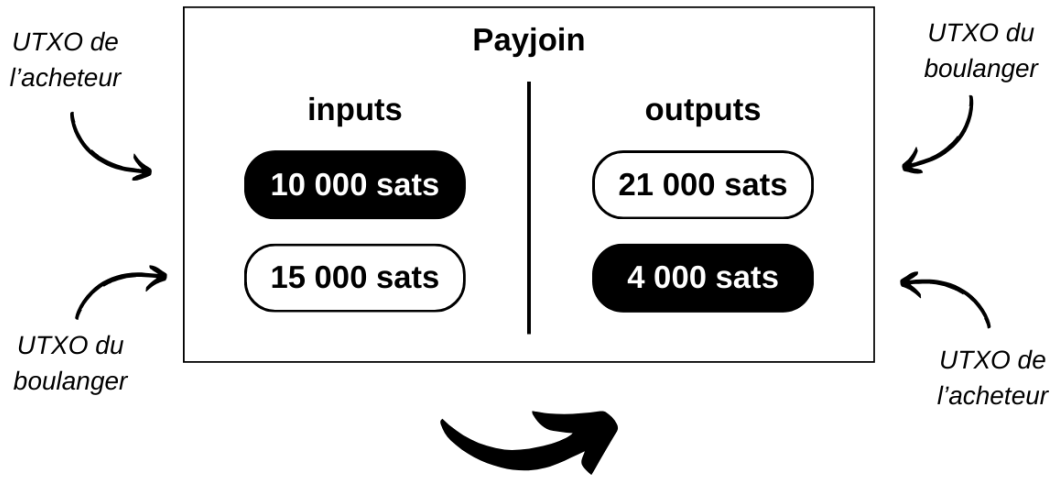
PATOSHI

Fait référence à un motif distinct de nonces et d'horodatages observés dans les blocs minés au cours des premiers mois de l'existence de Bitcoin. Ce modèle est soupçonné d'être attribuable à une seule entité ou individu, très probablement Satoshi Nakamoto lui-même, l'inventeur de Bitcoin. Le terme de « Patoshi » est un mot-valise combinant « pattern » et « Satoshi ». Plusieurs analyses des premiers blocs de Bitcoin ont révélé des motifs dans la façon dont les nonces et extra-nonces étaient choisis,

et comment les horodatages étaient définis. Ces motifs étaient suffisamment distincts pour suggérer qu'un mineur ou un groupe de mineurs particulier était responsable d'une grande proportion des blocs minés pendant cette période, tout en utilisant un client modifié. Sergio Demian Lerner, un chercheur en informatique, est crédité de la découverte de ce motif en 2013. Lerner a estimé que le mineur Patoshi avait miné environ 1,1 million de bitcoins. Cela a conduit à des spéculations généralisées sur les motivations, l'identité et les intentions actuelles du mineur Patoshi. Certains pensent que Patoshi était Satoshi lui-même, en train de miner des bitcoins pour soutenir et sécuriser le réseau naissant. Il est important de noter que, bien que le motif Patoshi soit largement accepté comme preuve d'une activité de minage précoce concentrée, il n'y a pas de confirmation définitive de l'identité derrière le Patoshi ou de ses intentions.

PAYJOIN

Structure spécifique de transaction Bitcoin qui permet d'améliorer la confidentialité des utilisateurs lors d'une dépense en collaborant avec le destinataire du paiement. La particularité du Payjoin réside dans sa capacité à générer une transaction qui paraît ordinaire à première vue, mais qui est en réalité un mini Coinjoin entre deux personnes. Pour cela, la structure de la transaction fait intervenir le destinataire du paiement dans les entrées aux côtés de l'expéditeur réel. Le destinataire inclut donc un paiement vers lui-même au milieu de la transaction qui permet elle-même de le payer. Par exemple, si vous achetez une baguette pour `6 000 sats` à l'aide d'un UTXO de `10 000 sats`, et que vous optez pour un Payjoin, votre boulanger ajoutera un UTXO de `15 000 sats` lui appartenant en entrée, qu'il récupèrera en intégralité en sortie, en plus de vos `6 000 sats`. La transaction Payjoin remplit deux objectifs. Tout d'abord, elle vise à induire en erreur un observateur extérieur en créant un leurre dans l'analyse de chaîne sur l'heuristique CIOH (*Common Input Ownership Heuristic*). Habituellement, lorsqu'une transaction sur la blockchain présente plusieurs entrées, on suppose que toutes ces entrées appartiennent vraisemblablement à une même entité. Ainsi, lorsqu'un analyste examine une transaction Payjoin, il est amené à croire que toutes les entrées proviennent d'une même personne. Toutefois, cette perception est erronée, car le destinataire du paiement contribue également aux entrées aux côtés du payeur réel. Ensuite, le Payjoin permet également de tromper un observateur extérieur sur le montant réel du paiement qui a été opéré. En examinant la structure de la transaction, l'analyste pourrait croire que le paiement est équivalent au montant d'une des sorties. En réalité, le montant du paiement ne correspond à aucun des outputs. Il est en fait la différence entre l'UTXO du destinataire en sortie et l'UTXO du destinataire en entrée. En ça, la transaction Payjoin rentre dans le domaine de la stéganographie. Elle permet de cacher le montant réel d'une transaction au sein d'une fausse transaction qui agit comme un leurre.



Le Payjoin est également parfois nommé « P2EP (Pay-to-End-Point) », « Stowaway » ou « transaction stéganographique ».

PAYNYM

Identifiant unique lié à un portefeuille Bitcoin qui implémente cette option. Les Paynyms sont disponibles sur Samourai Wallet et sur Sparrow Wallet. À l'origine, ces identifiants sont générés à partir du code de paiement du portefeuille, conformément au BIP47. Ils offrent ainsi la possibilité de se connecter avec un autre utilisateur afin de générer des adresses de réception vierges et uniques, et ce, sans nécessiter d'interaction directe. L'usage des Paynyms a ensuite été élargi pour soutenir diverses autres fonctionnalités de l'écosystème Samourai. Ils sont notamment employés pour permettre des échanges chiffrés sur le protocole de communication Soroban, afin de mettre en œuvre facilement des transactions collaboratives. Les Paynyms peuvent aussi servir à l'authentification sur des systèmes compatibles avec le protocole AUTH47. Chaque Paynym se caractérise par un identifiant unique et est représenté par une illustration de robot.

*Pour plus d'informations, voir la définition de **BIP47**.*

PBKDF2

`PBKDF2` est le sigle de *Password-Based Key Derivation Function 2*. C'est une méthode pour créer des clés cryptographiques à partir d'un mot de passe en utilisant une fonction de dérivation. Elle prend en entrée un mot de passe, un sel cryptographique, et applique de manière itérative une fonction prédéterminée (souvent une fonction de hachage comme `SHA256` ou un `HMAC`) sur ces données. Ce processus est répété de nombreuses fois afin de générer une clé cryptographique. Dans le contexte de Bitcoin, `PBKDF2` est utilisée en conjonction avec la fonction `HMAC-SHA512` pour créer la graine d'un portefeuille déterministe et hiérarchique (seed) à partir d'une phrase de récupération de 12 ou de 24 mots. Le sel cryptographique utilisé dans ce cas est la passphrase BIP39, et le nombre d'itérations est de `2048`.

PEER DISCOVERY

Processus par lequel les nœuds du réseau Bitcoin se connectent à d'autres nœuds pour obtenir des informations. Lorsqu'un nœud Bitcoin est lancé pour la première fois, il ne possède aucune information sur les autres nœuds du réseau. Pourtant, il doit établir des connexions pour se synchroniser sur la blockchain avec le plus de travail accumulé. Plusieurs mécanismes sont utilisés pour découvrir ces pairs, par ordre de priorité :

- Le nœud commence par consulter son fichier local `peers.dat` , qui stocke des informations sur les nœuds avec lesquels il a précédemment interagi. Si le nœud est nouveau, ce fichier sera vide, et le processus passera à l'étape suivante ;
- En l'absence d'informations dans le fichier `peers.dat` (ce qui est normal pour un nœud nouvellement lancé), le nœud effectue des requêtes DNS auprès des DNS seeds. Ces serveurs fournissent une liste d'adresses IP de nœuds à priori actifs pour établir des connexions. Les adresses des DNS seeds sont codées en dur dans le code de Bitcoin Core. Cette étape est généralement suffisante pour compléter la découverte des pairs ;
- Si les DNS seeds ne répondent pas dans les 60 secondes, le nœud peut alors se tourner vers les seed nodes. Ce sont des nœuds Bitcoin publics répertoriés dans une liste statique de près d'un millier d'entrées intégrée directement dans le code source de Bitcoin Core. Le nouveau nœud utilisera cette liste pour établir une première connexion au réseau et obtenir des adresses IP d'autres nœuds ;
- Dans le cas très peu probable où toutes les méthodes précédentes échouent, l'opérateur du nœud a toujours la possibilité d'ajouter manuellement des adresses IP de nœuds pour établir une première connexion.

PEERS.DAT

Nom du fichier de données utilisé par le logiciel Bitcoin Core pour stocker des informations sur les pairs (c'est-à-dire, les nœuds) du réseau avec lesquels le nœud de l'utilisateur a interagi ou peut potentiellement interagir. Il contient des détails comme les adresses IP, les numéros de ports et diverses métadonnées. Les nœuds présents dans cette liste sont par défaut les seed nodes, puis tous les autres nœuds découverts ou ajoutés manuellement. Ce fichier contient généralement une très grande liste de pairs dans laquelle le nœud pioche au hasard pour établir ses connections.

PERCOLATION

Fait référence à un modèle qui permet de comprendre la diffusion des informations (transactions et blocs) dans le réseau de nœuds Bitcoin. La théorie de la percolation est initialement un modèle mathématique et physique qui étudie le mouvement et la filtration de fluides à travers des matériaux poreux. Elle analyse comment, au-delà d'un certain seuil, un réseau connecté permet au fluide de s'écouler de manière continue à travers le matériau. On peut l'appliquer à des réseaux informatiques afin de voir comment les informations se diffusent en considérant les nœuds comme des sites pouvant être soit actifs, soit inactifs. Dans Bitcoin, les nœuds jouent ainsi le rôle des pores dans la théorie de la percolation. Chaque nœud actif reçoit et transmet l'information à d'autres nœuds qui vont soit continuer la transmission, soit la bloquer. La diffusion de certains types de transaction peut être analysée en termes de seuils de percolation, où un certain pourcentage de nœuds actifs est nécessaire pour atteindre un mineur qui l'inclura dans un bloc. Cette théorie permet d'avoir un cadre pour évaluer comment les changements dans le réseau, comme la modification des règles de standardisation par certains nœuds, affectent le mécanisme de propagation en cascade des transactions pour atteindre un mineur.

PÉRIODE DE MATURITÉ

Délai nécessaire avant qu'une récompense de bloc ne soit dépensable par le mineur qui l'a reçue. Cette période est fixée à 100 blocs suivant le bloc miné, soit 101 confirmations pour la transaction coinbase. Pendant ce laps de temps, les bitcoins nouvellement créés dans la récompense de bloc ne sont pas dépensables. Cette règle a pour but d'éviter les complications liées à l'utilisation de bitcoins issus d'une chaîne qui pourrait être ultérieurement rendue obsolète. En effet, il arrive que des blocs valides soient finalement invalidés si un autre bloc, à la même hauteur, est intégré dans une chaîne bénéficiant d'une plus grande preuve de travail. Ce phénomène, appelé réorganisation, aboutit à la création d'un « bloc orphelin » ou « bloc obsolète », privant ainsi le mineur des bitcoins contenus dans la coinbase du bloc abandonné. Si les bitcoins nouvellement créés étaient immédiatement dépensables, toute transaction les impliquant pourrait être annulée a posteriori, causant des pertes pour les détenteurs de ces bitcoins. Un tel scénario pourrait entraîner des annulations en série de transactions pourtant valides, affectant ainsi tous les utilisateurs impliqués dans cette chaîne de transactions. La période de maturité est donc un mécanisme de prévention contre ce risque. En imposant un délai de 100 blocs avant que les bitcoins nouvellement émis puissent être utilisés, on évite que des pièces issues de blocs finalement invalidés ne perturbent le système en circulant et en affectant d'autres transactions. La probabilité de voir survenir une réorganisation de 101 blocs est si faible qu'elle est considérée comme nulle.

PETIT-BOUTISTE

Format de stockage de données dans les systèmes informatiques où les octets les moins significatifs (les « petits bouts ») sont placés en premier dans l'ordre des adresses. Dans une séquence comportant plusieurs octets, l'octet ayant le plus petit poids (par exemple, les chiffres les plus à droite en hexadécimale) est stocké en premier.

En anglais, petit-boutiste se traduit par « Little-Endian ».

PHRASE DE RÉCUPÉRATION (MNÉMONIQUE)

Une phrase de récupération, également parfois nommée comme mnémonique, seed phrase, ou phrase secrète, est une séquence composée habituellement de 12 ou 24 mots, qui est générée de manière pseudo-aléatoire à partir d'une source d'entropie. La séquence pseudo-aléatoire est toujours complétée d'une somme de contrôle (checksum). La phrase mnémonique, conjointement avec une passphrase optionnelle, est utilisée pour dériver de façon déterministe l'intégralité des clés associées à un portefeuille HD (déterministe et hiérarchique). Cela signifie qu'à partir de cette phrase, il est possible de générer et de recréer déterministiquement l'ensemble des clés privées et publiques du portefeuille Bitcoin, et par conséquent d'accéder aux fonds qui y sont associés. La raison d'être de la phrase de récupération est de fournir un moyen de sauvegarde et de récupération des bitcoins qui est à la fois sécurisé et facile à utiliser. Il est impératif de conserver cette phrase en lieu sûr et de manière sécurisée, car toute personne en possession de cette phrase aurait accès aux fonds du portefeuille correspondant. Si elle est utilisée dans le cadre d'un portefeuille classique, et sans passphrase optionnelle, elle constitue souvent un SPOF (point de défaillance unique). La phrase de récupération est donc un encodage de la séquence pseudo aléatoire et de la checksum dans des mots du quotidien afin de faciliter sa notation et sa lisibilité par l'Homme. Elle est construite en fonction du standard BIP39, qui défini et ordonne une liste de 2048 mots utilisés pour cet encodage.

POOL

Service développé par Lightning Labs. C'est une sorte de marché pour les liquidités dans les canaux Lightning. Pool connecte les utilisateurs ayant besoin d'accès à la liquidité sur LN avec ceux qui ont

des capitaux à déployer sur le réseau. Les participants peuvent gagner des satoshis en ouvrant de nouveaux canaux pour ceux qui cherchent à recevoir des fonds sur Lightning pendant une période déterminée. Pool permet également de louer un canal pour accepter instantanément des paiements Bitcoin.

POOL DE MINAGE

Fait référence à un ensemble de mineurs qui collaborent en combinant leur puissance de calcul pour participer à la preuve de travail sur Bitcoin. Ce groupement en une seule organisation est une solution à la difficulté croissante de l'extraction de bitcoins, qui rend trop improbable pour un mineur individuel de rivaliser et de gagner des récompenses de manière stable. Les mineurs au sein d'un pool de minage contribuent avec leur matériel de calcul à la recherche d'une preuve de travail valide requise pour trouver un bloc. Lorsqu'un bloc est miné par le pool, la récompense, qui comprend les bitcoins nouvellement créés ainsi que les frais de transaction inclus dans le bloc, est distribuée parmi les membres du pool. Cette distribution est proportionnelle à la puissance de calcul que chaque mineur a contribué. En joignant leurs forces, les mineurs au sein d'un pool augmentent leurs chances de résoudre un bloc. Cela permet d'assurer une source de revenus plus régulière et prévisible par rapport au minage en solo, où un mineur peut ne pas gagner de récompense pendant de longues périodes. Cela permet de lisser les gains, et donc d'avoir d'une meilleure visibilité sur cette activité industrielle, notamment pour faire face aux différentes charges qu'elle induit.

> *En anglais, on dit « Mining pool ». Attention, la pool de minage ne doit pas être confondue avec la ferme de minage.*

POOL HOPPING

Désigne la pratique de certains mineurs consistant à changer fréquemment de pool de minage pour maximiser leurs gains. Ces mineurs passent d'une pool à une autre en fonction des variations de la rentabilité. Cette stratégie exploite les différences dans les méthodes de calcul des récompenses des pools. Le pool hopping peut déséquilibrer la distribution des récompenses au sein des pools et est généralement considéré comme une pratique déloyale dans la communauté.

PORTE DÉROBÉE (BACKDOOR)

Une backdoor est un mécanisme secret qui permet de disposer d'un accès privilégié à un système informatique, un logiciel, une fonction, un algorithme ou des données, sans passer par les procédures d'authentification ou de sécurité habituelles. À la différence d'une faille de sécurité, les portes dérobées sont introduites intentionnellement dans le code source par des développeurs malveillants. Elles peuvent être utilisées pour espionner, manipuler ou voler des informations sensibles.

> *Le terme de « porte dérobée » est assez peu utilisé en français. On préfère généralement employer directement la traduction anglaise qui est « backdoor ».*

PORTEFEUILLE

Outil logiciel spécialement conçu pour sécuriser et gérer les clés privées d'un utilisateur. Si le portefeuille est stocké et géré sur un dispositif logiciel lui-même installé sur une machine polyvalente, on parle alors de « portefeuille chaud ». En revanche, s'il est stocké dans un logiciel, lui-même installé sur un dispositif matériel dédié uniquement à cette tâche et non connecté à internet, on parle alors de « portefeuille froid ». Le portefeuille permet notamment d'utiliser les clés privées de l'utilisateur pour signer des transactions et ainsi remplir les conditions permettant la dépense des bitcoins.

En français, beaucoup utilisent directement la traduction anglaise « wallet » pour évoquer un portefeuille.

PORTEFEUILLE CHAUD (LOGICIEL)

Un portefeuille chaud (ou « hot wallet ») est un dispositif logiciel dédié à la sécurisation et à la gestion des clés privées d'un portefeuille Bitcoin. On parle de portefeuille chaud lorsque la phrase de récupération d'un portefeuille Bitcoin est conservée sur un appareil informatique, via un logiciel, qui n'est pas dédié uniquement à une utilisation de Bitcoin et qui est connecté directement à internet. Par exemple, l'application Samourai Wallet sur votre smartphone serait considérée comme un portefeuille chaud.

PORTEFEUILLE FROID

Un portefeuille froid, ou un hardware wallet, est un dispositif électronique dédié à la sécurisation et à la gestion des clés privées d'un portefeuille Bitcoin. Ces périphériques sont conçus pour procurer une sécurité renforcée par rapport aux portefeuilles logiciels qui résident sur des machines polyvalentes et directement connectées à internet. Les hardwares wallets stockent la phrase mnémonique hors ligne, sur un matériel qui dispose d'une infime surface d'attaque, ce qui l'isole des environnements potentiellement vulnérables. Lorsqu'une transaction est effectuée, le portefeuille matériel signe la transaction à l'intérieur du dispositif lui-même, sans exposer la clé privée à l'extérieur. Une fois la transaction signée, elle est transmise au réseau Bitcoin pour être confirmée et incluse dans la blockchain Bitcoin. Parmi les modèles de hardwares wallets les plus populaires, on peut citer : Ledger, Trezor, Coldcard, Passport, BitBox, Satochip, Jade ou encore SeedSigner (liste non exhaustive).

En anglais, portefeuille froid ou portefeuille matériel se traduit généralement par « Cold Wallet » ou « Hardware Wallet ».

POT (PAY ON TARGET)

Méthode de calcul de la rémunération des mineurs dans le contexte des pools de minage. POT est un système de rémunération variant selon la difficulté du travail envoyé à la pool plutôt que celle du travail fourni par la pool. Dans cette approche, la récompense pour chaque part soumise par un mineur est établie sur la difficulté de cette part spécifique. Cela signifie que les parts plus difficiles sont mieux récompensées que celles moins difficiles. C'est une méthode variante de PPS, mais en ajustant la récompense en fonction de la complexité réelle du travail accompli, POT implique beaucoup plus de variance pour le mineur.

PPLNS (PAY PER LAST N SHARES)

Méthode de calcul de la rémunération des mineurs dans le contexte des pools de minage. PPLNS récompense les mineurs en fonction de leur contribution en parts (shares) sur une période donnée. Dans PPLNS, les paiements sont effectués seulement lorsque la pool trouve un bloc et sont basés sur le nombre de parts soumises par le mineur par rapport au total des parts collectées pendant la période observée. Cette méthode favorise les mineurs constants et actifs sur le long terme, car elle décourage le « pool hopping » (changement fréquent de pool). La rémunération varie avec la probabilité de trouver un bloc, ce qui peut entraîner une baisse de la constance dans les revenus du mineur.

PPLNSG (PAY PER LAST N SHARES GROUPED)

Méthode de calcul de la rémunération des mineurs dans le contexte des pools de minage. PPLNSG fonctionne comme PPLNS, mais en regroupant les parts en équipes. Ces groupes de parts sont ensuite rémunérés ensemble.

PPS (PAY PER SHARE)

Méthode de calcul de la rémunération des mineurs dans le contexte des pools de minage. PPS est un système où les mineurs sont rémunérés pour chaque part (share) valide soumise, indépendamment du fait que le pool trouve ou non un bloc. Ils sont donc rémunérés sur la valeur attendue. Chaque part soumise est considérée comme une contribution au processus de minage et a une valeur fixe prédéterminée. Cette méthode offre une rémunération stable et prévisible pour les mineurs, car elle élimine la variabilité liée à la probabilité de trouver un bloc. Toutefois, elle est plus risquée pour les opérateurs de pool, car ils doivent payer les mineurs même lorsque aucun bloc n'est trouvé, absorbant ainsi le risque de variance. Contrairement à la méthode FPPS, PPS n'inclut pas les frais de transaction dans le calcul de la rémunération, mais seulement la subvention de bloc.

PRÉFIXES BINAIRES

Unités utilisées en informatique pour quantifier les multiples de tailles de données basées sur des puissances de 2. Contrairement aux préfixes du système métrique qui utilisent une base de 10, les préfixes binaires, tels que kibi (Ki), mebi (Mi), gibi (Gi), et tebi (Ti), multiplient par des puissances de 2 (2^10, 2^20, 2^30, 2^40 respectivement). Ces préfixes sont hérités des premières manières de mesurer la taille d'informations sur des ordinateurs. On les retrouve parfois dans Bitcoin, comme par exemple pour désigner la limite de taille des fichiers `BLOCKS/BLK?????.DAT` qui permettent de stocker les données brutes de la blockchain dans le logiciel Bitcoin Core. Ces derniers disposent ainsi d'une capacité maximale de 128 mébioctets (Mio), ce qui équivaut à un peu plus de 134 mégaoctets (Mo).

PREUVE DE TRAVAIL

Mécanisme de protection face aux attaques Sybil, qui se caractérisent par la multiplication de fausses identités, dans le but de prendre un avantage illégitime. Ainsi, la preuve de travail permet d'établir un coût marginal non négligeable à la multiplication des votes sur Bitcoin. La preuve de travail est à la base du protocole de consensus de Nakamoto, qui est le principe utilisé pour établir un accord sur une version unique du registre distribué entre les différents nœuds du réseau. Concrètement, la preuve de travail est la recherche d'une valeur qui, une fois passée dans une fonction mathématique aléatoire, donne un résultat inférieur à un nombre cible. Cette cible de la preuve de travail est ajustée tous les 2016 blocs par les nœuds. C'est ce que l'on appelle l'ajustement de la difficulté. On abaisse le nombre cible pour augmenter la difficulté de minage, ou on l'augmente pour baisser la difficulté, en fonction de l'évolution de la puissance de calcul déployée par les mineurs durant la période précédente. Ce travail effectué par les mineurs est récompensé à chaque bloc valide trouvé. Le mineur gagnant empoche une récompense pécuniaire, composée de la subvention de bloc (création de nouveaux bitcoins ex-nihilo), et des frais de transaction. Aujourd'hui, la difficulté de la preuve de travail sur Bitcoin est telle que le minage nécessite une grande puissance de calcul pour parvenir à gagner des blocs. En conséquence, il faut souvent disposer de puces électroniques spécialisées dans l'exécution de `SHA256`, c'est ce que l'on appelle un ASIC, et participer dans des pools de minage.

En anglais, on parle de « Proof-of-Work », parfois abrégé avec le sigle « PoW ».

PROOF-OF-WORK

Traduction anglaise de « Preuve de travail ».

*Pour plus d'informations, voir la définition de **PREUVE DE TRAVAIL**.*

PROP (PROPORTIONAL)

Méthode de calcul de la rémunération des mineurs dans le contexte des pools de minage. PROP répartit simplement la récompense de bloc parmi les mineurs proportionnellement à leur contribution en parts. Le calcul des parts débute au dernier bloc trouvé par la pool et termine lorsqu'un nouveau bloc est trouvé. Chaque nouveau bloc remet le compteur de parts à zéro. Cette méthode de rémunération permet de refléter directement les efforts par chacun.

PSEUDO-ALÉATOIRE

Cet adjectif est employé pour décrire une séquence de nombres qui, bien qu'étant le résultat d'un processus déterministe, affiche des caractéristiques qui se rapprochent de celles idéales d'une séquence véritablement aléatoire. La notion d'aléatoire idéal implique une absence totale de prévisibilité et de corrélation entre les éléments successifs. Un nombre pseudo-aléatoire est généré par un algorithme déterministe et est donc, en théorie, il est entièrement prévisible si l'on connaît l'état initial du générateur. Un générateur de nombres pseudo-aléatoires (« PRNG » en anglais, ou « GNPA » en français) est un algorithme utilisé pour produire de tels nombres. Il commence généralement à partir d'une valeur initiale, ou « graine », et applique ensuite une série de transformations mathématiques pour produire la suite de nombres. Du fait de cette déterminabilité, il est crucial pour la sécurité cryptographique que la graine initiale reste secrète. Les suites pseudo-aléatoires sont largement utilisées dans divers domaines, notamment la cryptographie, car elles manifestent un comportement apparemment aléatoire qui suffit pour de nombreuses applications. L'évaluation de la qualité d'un PRNG repose sur la mesure dans laquelle sa sortie se rapproche d'un véritable aléa en termes de distribution, de corrélations et d'autres propriétés statistiques. Dans le cadre de Bitcoin, les nombres pseudo-aléatoires sont utilisés pour produire des clés privées, ou bien pour produire une graine pour les portefeuilles déterministes et hiérarchique.

PULL REQUEST

Dans le cadre de Github et d'autres plateformes d'hébergement de code, une Pull Request représente une demande faite par un contributeur pour intégrer ses modifications d'une branche de son fork à une branche du dépôt principal. Elle déclenche une révision de code et une discussion avant que les changements ne soient potentiellement fusionnés (merge). Ce processus est très utilisé dans le développement des implémentations de nœuds Bitcoin, notamment Bitcoin Core.

Le terme de « Pull Request » est souvent abrégé par le sigle « PR ».

PYTHON

Langage de programmation de haut niveau, connu pour sa syntaxe claire et sa lisibilité. Python est polyvalent, utilisé dans le développement web, l'analyse de données, l'intelligence artificielle, la science des données et l'automatisation. Il est apprécié pour sa simplicité et sa large communauté.

Q

QUBIT

Unité d'information de base sur un ordinateur quantique. Ces qubits peuvent prendre la valeur de 0, la valeur de 1, ou bien une superposition du 0 et du 1. En utilisant cette superposition d'états avec d'autres phénomènes quantiques tels que l'intrication et l'interférence quantique, un ordinateur quantique peut paralléliser les processus de calculs, et donc résoudre certains problèmes spécifiques beaucoup plus rapidement.

R

RACINE DE MERKLE

Condensat ou « top hash » d'un arbre de Merkle, qui représente un résumé de toutes les informations présentes dans l'arbre. Un arbre de Merkle est une structure d'accumulateur cryptographique, parfois également nommée « arbre de hachage ». Dans le cadre de Bitcoin, des arbres de Merkle sont utilisés pour organiser les transactions dans un bloc et pour faciliter la vérification rapide de l'inclusion d'une transaction spécifique. Ainsi, dans les blocs de Bitcoin, la racine de Merkle est obtenue en hachant de manière successive les transactions par paires jusqu'à ce qu'il ne reste qu'un seul hachage (la racine de Merkle). Cette dernière est ensuite incluse dans l'en-tête du bloc correspondant. On retrouve également cette structure dans UTREEXO, une structure permettant de condenser l'UTXO set des nœuds, et dans le MAST Taproot.

*Pour plus d'informations, voir la définition d'**ARBRE DE MERKLE***

RBF (REPLACE-BY-FEE)

Mécanisme transactionnel permettant à l'expéditeur de remplacer une transaction par une autre avec des frais plus élevés, afin d'accélérer la confirmation de celle-ci. Si une transaction avec des frais trop faibles reste bloquée, l'expéditeur peut utiliser Replace-By-Fee (remplacement par les frais) pour augmenter les frais et privilégier sa transaction de remplacement dans les mempools. RBF est applicable tant que la transaction est dans les mempools ; une fois dans un bloc, elle ne peut plus être remplacée. Lors de l'envoi initial, la transaction doit spécifier sa disponibilité à être remplacée en ajustant la valeur de `nSequence` à une valeur inférieure à `0xfffffffe`. C'est ce que l'on appelle un « flag » RBF. Ce paramètre signale la possibilité de mise à jour de la transaction après sa diffusion, offrant ainsi la possibilité de faire un RBF. Cependant, il est parfois possible de remplacer une transaction n'ayant pas signalé RBF. Les nœuds utilisant le paramètre de configuration `mempoolfullrbf=1` acceptent ce remplacement même si RBF n'a pas été signalé initialement.

RÉCOMPENSE DE BLOC

Total des bitcoins récupérés par un mineur lorsqu'il trouve un bloc valide sur Bitcoin. Cette récompense est composée de deux éléments : la subvention de bloc et les frais de transaction. La subvention de bloc est une quantité fixe de bitcoins que le mineur peut créer ex nihilo. Cette quantité diminue progressivement au fil des halvings. Les frais de transaction sont les frais cumulés payés par les utilisateurs pour effectuer les transactions incluses dans le bloc miné. Les frais sont également des bitcoins « créés » par le mineur, mais leur quantité est limitée au montant des bitcoins « détruits » dans les transactions. En effet, les frais d'une transaction représentent la différence entre le total des entrées et le total des sorties. La récompense de bloc est distribuée au sein d'une transaction spécifique que l'on appelle « coinbase ». Les bitcoins qui en sont extraits sont automatiquement bloqués durant une période de 100 blocs. C'est ce que l'on appelle la période de maturité.

La traduction anglaise est « Block Reward ».

RÉCURSIF (COVENANT)

Un covenant récursif sur Bitcoin est un type de contrat intelligent qui impose des conditions non seulement sur la transaction actuelle mais aussi sur les transactions futures qui dépensent les sorties de cette transaction. Cela permet de créer des chaînes de transactions où chacune doit respecter certaines règles définies par la première de la chaîne. La récursivité crée une séquence de transactions où chacune hérite des restrictions de sa transaction parent. Cela permettrait d'établir un contrôle complexe et à long terme sur la manière dont les bitcoins peuvent être dépensés, mais cela introduirait également des risques au niveau de la liberté de dépense et de la fongibilité. Pour résumer,

un covenant non récursif se limitera uniquement à la transaction qui succède immédiatement à celle qui a établi les règles. Et au contraire, un covenant récursif aura la capacité d'imposer des conditions spécifiques à un bitcoin de manière indéfinie. Les transactions pourront se succéder, mais le bitcoin en question conservera toujours les conditions initiales qui lui sont attachées. De manière plus générale, en informatique, ce que l'on appelle la « récursivité » est la capacité d'une fonction à s'appeler elle-même, ce qui crée une sorte de mise en abyme.

*Pour plus d'informations, voir la définition de **COVENANT**.*

REDEEMSCRIPT

Script qui défini les conditions spécifiques que doivent remplir les inputs pour débloquer les fonds associés à un output P2SH. Dans un UTXO P2SH, le `scriptPubKey` contient le hachage du `RedeemScript`. Lorsqu'une transaction souhaite dépenser cet UTXO en entrée, elle doit fournir le `RedeemScript` en clair qui correspond au hachage contenu dans le `scriptPubKey`. Le `RedeemScript` est donc donné dans le `scriptSig` de l'input, en plus des autres éléments nécessaires pour satisfaire les conditions du script, comme les signatures ou les clés publiques. Cette structure encapsulée garantit que les détails des conditions de dépense restent cachés jusqu'à ce que les bitcoins soient effectivement dépensés. On l'utilise notamment pour les portefeuilles multisignatures Legacy.

RÈGLES DE CONSENSUS

Règles fondamentales dans Bitcoin, assurant l'intégrité du réseau en dictant les critères de validation des transactions et des blocs. Une transaction qui ne suit pas les règles de consensus ne peut pas être minée dans un bloc valide. Par exemple, une transaction qui comporterait une signature invalide pour une de ses entrées ne pourrait pas être incluse dans un bloc. On différencie alors les règles de consensus et les règle de standardisation.

RÈGLES DE STANDARDISATION

Règles adoptées individuellement par chaque nœud Bitcoin, en plus des règles de consensus, pour définir la structure des transactions non confirmées qu'il accepte dans sa mempool et diffuse à ses pairs. Ces règles sont donc configurées et exécutées en local par chaque nœud et peuvent varier d'un nœud à l'autre. Elles s'appliquent exclusivement sur les transactions non confirmées. Ainsi, un nœud n'acceptera une transaction qu'il jugerait non standard que si celle-ci est déjà incluse dans un bloc valide. Notons que la majorité des nœuds laissent les configurations par défaut telles que préétablies dans Bitcoin Core, engendrant de fait une homogénéité des règles de standardisation à travers le réseau. Une transaction qui, bien que conforme aux règles de consensus, ne respecte pas ces règles de standardisation, aura des difficultés à se propager sur le réseau. Elle pourra toutefois être incluse dans un bloc valide si jamais elle atteint un mineur. Dans la pratique, ces transactions, qualifiées de non standard, sont souvent transmise directement à un mineur par des voies externes au réseau pair-à-pair de Bitcoin. C'est souvent le seul moyen pour confirmer ce type de transaction. Par exemple, une transaction qui n'alloue aucun frais est à la fois valide selon les règles de consensus et non standard car la politique par défaut de Bitcoin Core pour le paramètre `minRelayTxFee` est de `0.00001` (en BTC/kB).

On parle également parfois de « règles de mempool ».

RÉORGANISATION

Se réfère à un phénomène où la blockchain subit une modification de sa structure à cause de l'existence de blocs concurrents à une même hauteur. Cela survient lorsqu'une portion de la chaîne

de blocs est remplacée par une autre chaîne ayant une quantité de preuve de travail accumulée plus importante. Ces réorganisations font partie du fonctionnement naturel de Bitcoin, où différents mineurs peuvent trouver de nouveaux blocs presque simultanément, venant ainsi couper le réseau Bitcoin en deux. Dans de tels cas, le réseau peut se diviser temporairement en chaînes concurrentes. Finalement, lorsque l'une de ces chaînes accumule plus de preuve de travail, les autres chaînes sont abandonnées par les nœuds, et leurs blocs deviennent ce que l'on appelle des « blocs périmés ». Ce processus de remplacement d'une chaîne par une autre est la réorganisation. Les réorganisations peuvent avoir diverses conséquences. Tout d'abord, si un utilisateur avait une transaction confirmée dans un bloc qui s'avère être périmé, mais que celle-ci ne se retrouve pas dans la chaîne finalement valide, alors sa transaction redevient non confirmée. C'est pour cette raison que l'on vous conseille de toujours attendre au moins 6 confirmations pour considérer une transaction comme réellement immuable. Passé 6 blocs de profondeur, les réorganisations sont tellement improbables que la chance qu'il y en ait une peut être considérée comme nulle. Ensuite, au niveau du système global, les réorganisations impliquent un gaspillage de la puissance de calcul des mineurs. En effet, lorsqu'une division intervient, une partie des mineurs seront sur la chaîne `A`, et une autre partie sur la chaîne `B`. Si la chaîne `B` est finalement abandonnée lors d'une réorganisation, alors toute la puissance de calcul déployée par les mineurs sur cette chaîne est par définition gaspillée. Si il y a trop de réorganisations sur le réseau Bitcoin, la sécurité globale de celui-ci est donc réduite. C'est notamment pour cette raison, en partie, que l'augmentation de la taille des blocs ou la réduction de l'intervalle entre chaque bloc (10 minutes) peuvent être dangereuses.

> *Certains bitcoiners préfèrent parler de « bloc orphelin » pour désigner un bloc périmé. Aussi, dans le langage courant, on parle d'une « réorg » pour désigner une « réorganisation ». Le terme de « réorganisation » est un anglicisme. Pour être plus juste, on pourrait parler d'une « resynchronisation.*

RÉSEAU BITCOIN

Désigne l'infrastructure globale du système Bitcoin. Le réseau est constitué de l'ensemble des nœuds (ordinateurs) qui exécutent un logiciel implémentant le protocole Bitcoin, et qui se connectent à leurs pairs. Chaque nœud communique en pair-à-pair avec les autres nœuds afin de télécharger et de vérifier la blockchain, de vérifier et de diffuser les nouveaux blocs, et de vérifier et de diffuser les nouvelles transactions.

RÉSISTANCE AU PARTITIONNEMENT

Capacité du réseau Bitcoin à rester unifier et à maintenir le consensus entre les utilisateurs, en maintenant des connexions et en évitant la séparation de certains nœuds du reste du réseau, malgré les tentatives de le fragmenter. Pour qu'un nœud demeure en consensus avec le réseau, il doit maintenir au moins une connexion active avec un ensemble de pairs partageant les mêmes règles de consensus.

> *En anglais, on dit « Partition Resistance ».*

RESYNCHRONISATION

Synonyme plus juste de « réorganisation » bien que peu employé.

> *Pour plus d'informations, voir la définition de **RÉORGANISATION**.*

RÉUTILISATION D'ADRESSE

Se réfère à la pratique d'utiliser une même adresse de réception pour bloquer plusieurs UTXO, parfois au sein de plusieurs transactions différentes. Les bitcoins sont généralement bloqués à l'aide d'une paire de clés cryptographique qui correspond à une adresse unique. Puisque la blockchain est publique, il est facile de pouvoir consulter quelles adresses sont associées à combien de bitcoins. En cas de réutilisation d'une même adresse pour plusieurs paiements, on peut raisonnablement imaginer que tous les UTXO associés appartiennent à une même entité. La réutilisation d'adresse pause donc un problème pour la vie privée de l'utilisateur. Elle permet de faire des liens déterministes entre plusieurs transactions et plusieurs UTXO, ainsi que de perpétuer un traçage de fonds on-chain. Satoshi Nakamoto évoquait déjà ce problème dans son White Paper :

> « *En guise de pare-feu additionnel, une nouvelle paire de clés pourrait être utilisée pour chaque transaction afin de les garder non liées à un propriétaire commun.* » - Nakamoto, S. (2008). "Bitcoin: A Peer-to-Peer Electronic Cash System". Consulté à l'adresse https://bitcoin.org/bitcoin.pdf.

Pour préserver au minimum sa vie privée, il est vivement conseillé de n'utiliser chaque adresse de réception qu'une seule fois. À chaque nouveau paiement, il convient de générer une nouvelle adresse. Pour les outputs de change, il faut également utiliser une adresse vierge. Heureusement, grâce aux portefeuille déterministes et hiérarchiques, il est devenu très facile d'utiliser une multitudes d'adresses. Toutes les paires de clés associées à un portefeuille peuvent être facilement régénérées à partir de la graine. C'est d'ailleurs pour cette raison que les logiciels de portefeuille réputés vous génèrent toujours une nouvelle adresse différente lorsque vous cliquez sur le bouton « Recevoir ».

En anglais, on dit « Address Reuse ».

RICOCHET

Technique consistant à réaliser plusieurs transactions fictives vers soi-même pour simuler un transfert de propriété des bitcoins. Le Ricochet permet d'estomper les spécificités pouvant compromettre la fongibilité d'une pièce Bitcoin. Par exemple, si vous réalisez un coinjoin, votre pièce en sortie de mix sera identifiée comme telle. Cette étiquette de « *pièce issue d'un coinjoin* » peut affecter la fongibilité d'un UTXO. Des entités réglementées, telles que les plateformes d'échange, peuvent refuser d'accepter un UTXO ayant subi un coinjoin, voire exiger des explications de la part de son propriétaire, avec le risque de voir son compte bloqué ou ses fonds gelés. Dans certains cas, la plateforme peut même signaler votre comportement aux autorités étatiques. C'est là que la méthode du Ricochet entre en jeu. Pour estomper l'empreinte laissée par un coinjoin, Ricochet exécute quatre transactions successives où l'utilisateur transfère ses fonds à lui-même sur des adresses différentes. Après cet enchaînement de transactions, l'outil Ricochet achemine finalement les bitcoins vers leur destination finale, comme par exemple une plateforme d'échange. L'objectif est de créer de la distance entre la transaction coinjoin originale et l'acte de dépense final. De cette manière, les outils d'analyse de chaîne vont penser qu'il y a vraisemblablement eu un transfert de propriété après le coinjoin, et qu'il est donc inutile d'entamer des actions à l'encontre de l'émetteur. Le cas d'utilisation le plus courant de Ricochet se présente quand il est nécessaire de dissimuler une participation antérieure à un coinjoin sur un UTXO, notamment pour éviter d'être la cible des politiques LCB/FT des plateformes régulées ou des blacklists. L'outil Ricochet est disponible sur le protefeuille Samourai Wallet.

RIPEMD160

Acronyme de *Research and development in Advanced Communications technologies in Europe Integrity Primitives Evaluation Message Digest 160*, est une fonction de hachage cryptographique qui

génère un condensat de 160 bits à partir d'une entrée arbitraire. Elle est utilisée sur Bitcoin pour transformer une clé publique en une adresse de réception. Le processus implique d'abord l'application de la fonction de hachage `SHA256` sur la clé publique, suivie de l'application de `RIPEMD160` sur le résultat. Cette combinaison de deux fonctions de hachage distinctes est connue sous le nom de `HASH160` dans le contexte de Bitcoin. `RIPEMD160` est également utilisé dans les portefeuilles déterministes et hiérarchiques pour calculer des empreintes de clés. On utilise notamment `HASH160` pour calculer l'empreinte d'une clé parent, ensuite incluse dans les métadonnées d'une clé étendue (xpub).

*Pour plus d'informations, voir la définition de **FONCTION DE HACHAGE CRYPTOGRAPHIQUE**.*

RPC (REMOTE PROCEDURE CALL)

Protocole informatique permettant à un programme d'exécuter une procédure sur un autre ordinateur distant, comme si elle était exécutée localement. Spécifiquement dans le cadre de Bitcoin, on l'utilise pour permettre aux applications d'interagir avec bitcoind. Il peut être utilisé pour exécuter des commandes sur un nœud Bitcoin, telles que l'envoi de transactions, la gestion de portefeuilles ou encore l'accès à des informations sur la blockchain. La sécurité de cette interaction est assurée par une authentification via un fichier `.cookie` ou des identifiants, afin que seuls les clients autorisés puissent effectuer des RPC sur le nœud.

En français, on peut le traduire par « Appel de procédure à distance ».

RPOW (REUSABLE PROOFS OF WORK)

Système de monnaie électronique par transfert de jetons basés sur des preuves de travail, développé et mis en œuvre par Hal Finney en 2004. RPoW se positionnait comme une amélioration des concepts théoriques de b-money et bit gold. Contrairement à ces derniers, RPoW a effectivement vu le jour et a été lancé. RPoW aurait pu prendre la place qu'occupe actuellement Bitcoin. C'était le projet le plus aboutit de monnaie électronique avant l'invention de Satoshi. Toutefois, Bitcoin surpasse RPoW en résolvant deux problèmes critiques. Premièrement, Bitcoin a introduit un ajustement automatique de la difficulté de minage, un mécanisme absent dans RPoW, évitant ainsi l'inflation due à l'augmentation des capacités de minage et au nombre croissant de mineurs. Deuxièmement, contrairement à la dépendance de RPoW aux serveurs centraux, Bitcoin a instauré un mécanisme de consensus décentralisé. Ce mécanisme repose sur le principe que les nœuds se synchronisent sur la chaîne avec le plus de travail accumulé, éliminant ainsi la nécessité de serveurs connus. RPoW n'a jamais reçu le soutient nécessaire pour émerger et être adopté par le grand public. Contrairement à b-money et bit gold, Satoshi Nakamoto n'a jamais cité RPoW, alors que ce système était sûrement ce qui ressemblait le plus à son invention.

RSMPPS (RECENT SHARED MAXIMUM PAY PER SHARE)

Méthode de calcul de la rémunération des mineurs dans le contexte des pools de minage. RSMPPS est similaire à SMPPS, mais avec une priorité accordée aux mineurs ayant contribué récemment. Cette méthode vise à récompenser les contributions actuelles en augmentant la valeur des parts soumises dans les tours de minage les plus récents, favorisant ainsi les mineurs qui restent actifs.

RUST

Langage de programmation moderne axé sur la sécurité et la performance. Conçu pour éviter les erreurs courantes de programmation, Rust est utilisé dans les systèmes embarqués, les applications

Web, et pour le développement de logiciels nécessitant de hautes performances et une grande fiabilité. Ce langage est de plus en plus populaire dans l'environnement de Bitcoin.

RUST-LIGHTNING

Bibliothèque Lightning développée en Rust par la communauté Rust Bitcoin en collaboration avec Square. Rust-Lightning fournit une implémentation de Lightning. Elle sert de base au Lightning Development Kit (LDK).

S

SAMOURAI WALLET

Logiciel de portefeuille Bitcoin pour appareils mobiles Android axé sur la confidentialité. Il offre des fonctionnalités avancées telles que les coinjoins Whirlpool, Stonewall, StonewallX2, Ricochet ou encore Stowaway (payjoin). Samourai implémente également de nombreuses protections pour aider l'utilisateur à protéger sa vie privée face à l'analyse de chaîne.

SATOSHI (SAT)

Le satoshi, souvent abrégé en « sat », est la plus petite subdivision du bitcoin qui peut être enregistrée sur la blockchain. Il est nommé en l'honneur de l'inventeur de Bitcoin, Satoshi Nakamoto. Un seul Bitcoin se divise en 100 000 000 sats, ce qui signifie qu'un satoshi équivant à 0,00000001 bitcoin. En raison de sa petite valeur unitaire, le sat est souvent utilisé pour établir des prix, en particulier dans les petites transactions. Son utilisation est souvent préférée au btc sur le Lightning Network.

SATOSHI NAKAMOTO

Pseudonyme de la personne ou du groupe qui a créé Bitcoin et écrit son livre blanc original en 2008 (White Paper). Nakamoto, qui a communiqué uniquement en ligne, a finalement disparu de la scène publique en 2011.

SCALA

Langage de programmation conçu pour être concis, combinant programmation fonctionnelle et orientée objet. Scala est souvent utilisé pour les applications d'entreprise, le développement de systèmes complexes et le traitement de données.

SCHNORR (PROTOCOLE)

Le protocole de Schnorr est un algorithme de signatures électroniques établi sur la cryptographie sur les courbes elliptiques (ECC). Il est utilisé sur Bitcoin pour dériver une clé publique à partir d'une clé privée et pour signer une transaction avec une clé privée. Sur Bitcoin, tout comme ECDSA, Schnorr est établi sur l'exploitation de la courbe elliptique `secp256k1`, caractérisée par l'équation : $y^2 = x^3 + 7$. Le protocole de signature de Schnorr est implémenté dans le protocole Bitcoin depuis Novembre 2021 avec l'activation de la mise à jour de Taproot.

SCORE (SCORE BASED METHOD)

Méthode de calcul de la rémunération des mineurs dans le contexte des pools de minage. Ce système de récompense est proportionnel, mais pondéré par le moment auquel la part est soumise. SCORE valorise les parts en fonction du temps écoulé depuis le début du cycle de minage. Plus une part est soumise tardivement dans le cycle, plus sa valeur est élevée. Cette méthode permet d'inciter les mineurs à rester, car à chaque arrêt du minage, le mineur voit son score stagner alors que celui des autres augmente de plus en plus rapidement.

Cette méthode est parfois également nommée « Bitcoin Pooled Mining » (BPM).

SCRIPT

Langage de programmation à pile utilisé pour établir des conditions de dépense, et donc, indirectement, sécuriser des bitcoins. Script est essentiellement une liste d'instructions, composée

d'opérateurs logiques et de commandes pour manipuler la pile (stack). Il se matérialise par l'utilisation d'OPcodes qui donnent des instructions spécifiques qui sont exécutées par les nœuds du réseau lors de l'ajout d'une transaction à la blockchain. Script est un langage non-Turing complet. Il peut-être catégorisé comme un langage de niveau intermédiaire (presque bas niveau) inspiré du Forth.

SCRIPTLESS SCRIPTS

Concept initialement développé par Andrew Poelstra qui permet l'exécution de contrats intelligents sans exposer explicitement la logique du contrat sur la blockchain Bitcoin. Comme le suggère l'appellation « script sans script », l'idée repose sur l'exécution de scripts (ou de contrats) sans recourir explicitement à des scripts. Ces contrats exploitent les propriétés des signatures de Schnorr qui permettent l'usage des Adaptors Signatures, notamment pour réaliser des Atomic Swaps. Les conditions du contrat sont appliquées et exécutées off-chain par les parties impliquées, qui sont les seules à en connaître les termes. Contrairement aux contrats intelligents classiques, les Scriptless Scripts minimisent leur empreinte sur la blockchain, réduisant ainsi le coût de l'opération. Ces contrats sont aussi plus discrets que les contrats intelligents classiques, qui laissent des traces sur la blockchain. Ils ressemblent donc à des transactions ordinaires, ce qui accroît leur potentiel d'anonymat.

SCRIPTPUBKEY

Script situé dans la partie sortie (output) d'une transaction Bitcoin qui définit les conditions sous lesquelles l'UTXO associé peut être dépensé. Ce script permet donc de sécuriser des bitcoins. Dans sa forme la plus courante, le `scriptPubKey` contient une condition qui exige que la prochaine transaction fournisse une preuve de possession de la clé privée correspondant à une adresse Bitcoin spécifiée. C'est souvent réalisé par un script qui demande une signature correspondant à la clé publique associée à l'adresse utilisée pour sécuriser ces fonds. Lorsqu'une transaction tente d'utiliser cet UTXO en entrée (input), elle doit fournir un `scriptSig` qui, une fois associé avec le `scriptPubKey`, satisfait les conditions posées et produit un script valide. Cela implique généralement de prouver la possession de la clé privée associée grâce à une signature. Par exemple, voici un `scriptPubKey` P2PKH classique : `OP_DUP OP_HASH160 OP_PUSHBYTES_20 <adresse> OP_EQUALVERIFY OP_CHECKSIG`. Le `scriptSig` correspondant serait : `<signature> <clé publique>`.

> *Pour nommé ce script, on parle également parfois d'un « locking script » ou « script de verrouillage » en français.*

SCRIPTSIG

Élément dans une transaction Bitcoin, situé dans les entrées (input). Le `scriptSig` fournit les données nécessaires pour satisfaire les conditions posées par le `scriptPubKey` de la transaction précédente dont les fonds sont dépensés. Il joue donc un rôle complémentaire au `scriptPubKey`. Typiquement, le `scriptSig` contient une signature numérique et une clé publique. La signature est générée par le propriétaire des bitcoins à l'aide de sa clé privée et prouve qu'il a l'autorisation de dépenser l'UTXO. Dans ce cas, le `scriptSig` démontre que le détenteur de l'input possède la clé privée correspondant à la clé publique associée à l'adresse spécifiée dans le `scriptPubKey` de la transaction sortante précédente. Lorsque la transaction est vérifiée, les données du `scriptSig` sont exécutées dans le `scriptPubKey` correspondant. Si le résultat est valide, cela signifie que les conditions de dépense des fonds ont été remplies. Si toutes les entrées de la transaction fournissent un `scriptSig` qui valide leur `scriptPubKey`, la transaction est valide et pourra être ajoutée à un bloc pour son exécution. Par exemple, voici un `scriptSig` P2PKH classique : `<signature>`

`<clé publique>`. Le `scriptPubKey` correspondant serait : `OP_DUP OP_HASH160 OP_PUSHBYTES_20 <adresse> OP_EQUALVERIFY OP_CHECKSIG`.

Le scriptSig est également parfois nommé « unlocking script » ou « script de déverrouillage » en français.

SCRIPTWITNESS

Élément dans les entrées de transactions SegWit qui contient les signatures et les clés publiques nécessaires pour déverrouiller les bitcoins envoyés dans la transaction. Semblable au `ScriptSig` des transactions Legacy, le `ScriptWitness` n'est toutefois pas placé au même endroit. En effet, c'est cette partie, que l'on appelle le « témoin » (« witness » en anglais), qui est déplacée dans une base de données séparée afin de résoudre le problème de la malléabilité des transactions. Chaque input SegWit possède son propre `ScriptWitness`, et tous les `ScriptWitness` forment ensemble le champ `Witness` de la transaction.

Attention de ne pas confondre le ScriptWitness avec le WitnessScript. Tandis que le ScriptWitness contient les données de témoin de tout input SegWit, le WitnessScript définit les conditions de dépense d'un UTXO P2WSH ou P2SH-P2WSH et constitue un script à part entière, à la manière du redeemScript pour une sortie P2SH.

SDK (SOFTWARE DEVELOPMENT KIT)

Ensemble d'outils logiciels fournissant les ressources nécessaires aux développeurs pour créer des applications sur une plateforme spécifique. Un SDK inclut des bibliothèques, des guides de développement, des exemples de code ou encore des processus de compilation. Les SDK facilitent et accélèrent le développement en offrant des modules réutilisables. Sur Bitcoin, il existe le BDK (*Bitcoin Dev Kit*) et le LDK (*Lightning Dev Kit*).

En anglais, les SDK sont également parfois appelés « devkit ». Pour plus d'informations, voir les définitions de BDK et de LDK.

SECP256K1

Nom donné à une courbe elliptique spécifique utilisée dans le cadre du protocole Bitcoin pour l'implémentation des algorithmes de signatures numériques ECDSA (*Elliptic Curve Digital Signature Algorithm*) et Schnorr. La courbe `secp256k1` a été choisie par l'inventeur de Bitcoin, Satoshi Nakamoto. Elle présente certaines propriétés intéressantes, notamment des avantages en termes de performance. L'utilisation de `secp256k1` sur Bitcoin implique que chaque clé privée (un nombre aléatoire de 256 bits) est mappée à une clé publique correspondante par multiplication de la clé privée par le point générateur de la courbe `secp256k1`. Cette opération est facile à réaliser dans un sens, mais pratiquement impossible à inverser, ce qui constitue la base de la sécurité des signatures numériques sur Bitcoin. La courbe `secp256k1` est spécifiée par l'équation de la courbe elliptique $y^2 = x^3 + 7$, ce qui signifie qu'elle a des coefficients a égal à `0` et b égal à `7` dans la forme générale de l'équation d'une courbe elliptique $y^2 = x^3 + ax + b$. `Secp256k1` est définie sur un corps fini dont l'ordre est un nombre premier très grand, spécifiquement $p = 2^{256} - 2^{32} - 977$. La courbe a également un ordre de groupe, qui est le nombre de points distincts sur la courbe, un point générateur (ou point G) prédéfini, qui est utilisé dans les opérations de cryptographie pour générer des paires de clés, et un cofacteur qui est égal à `1`.

« SEC » désigne « Standards for Efficient Cryptography ». « P256 » désigne le fait que la courbe est définie sur un corps $\mathbb{Z}_p$ où p est un nombre premier de 256 bits. « K » désigne

le nom de son inventeur, Neal Koblitz. Enfin, « 1 » désigne que c'est la première version de cette courbe.

SEED NODES

Liste statique de nœuds Bitcoin publics, intégrée directement dans le code source de Bitcoin Core (bitcoin/src/chainparamsseeds.h). Cette liste sert de points de connexion pour les nouveaux nœuds Bitcoin qui rejoignent le réseau, mais elle n'est utilisée que si les DNS seeds ne fournissent pas de réponse dans les 60 secondes suivant leur sollicitation. Dans ce cas, le nouveau nœud Bitcoin se connectera à ces seed nodes pour établir une première connexion au réseau et demander des adresses IP d'autres nœuds. L'objectif final est d'acquérir les informations nécessaires pour effectuer l'IBD et se synchroniser avec la chaîne qui a le plus de travail accumulé. La liste des seed nodes comprend près de 1000 nœuds, identifiés conformément à la norme établie par le BIP155. Ainsi, les seed nodes représentent la troisième méthode de connexion au réseau pour un nœud Bitcoin, après l'éventuelle utilisation du fichier peers.dat, créé par le nœud lui-même, et la sollicitation des DNS seeds.

Attention, les seed nodes ne doivent pas être confondus avec les « DNS seeds », qui sont eux la deuxième manière d'établir des connexions. Pour plus d'informations, voir la définition de ***DNS SEEDS****.*

SEGWIT

SegWit, acronyme pour « Segregated Witness » (Témoin Séparé), est une mise à jour du protocole Bitcoin, introduite en août 2017. Elle vise à résoudre plusieurs problèmes techniques, dont la question de la capacité transactionnelle du réseau, le problème de malléabilité des transactions et la facilitation des modifications futures du protocole. Ce Soft Fork modifie la structure des transactions en déplaçant les données de signature de la transaction vers un répertoire séparé. Concrètement, avec SegWit, les signatures sont retirées du bloc principal et insérées dans une structure de données distincte à la fin du bloc, ce sont les témoins (witness). Cette séparation permet d'augmenter la capacité de chaque bloc sans modifier la taille maximale des blocs elle-même, qui est de 1 Mo sur Bitcoin. Cette modification résout également le problème de la malléabilité des transactions. Avant SegWit, les signatures pouvaient être modifiées avant qu'une transaction ne soit confirmée, ce qui changeait l'identifiant de la transaction. Cela rendait difficile la construction de transactions complexes, car une transaction non confirmée pouvait voir son identifiant changer. En séparant les signatures, SegWit rend les transactions non malléables, car tout changement dans les signatures n'affecte plus l'identifiant de la transaction (TXID), mais uniquement l'identifiant du témoin (WTXID). En résolvant le problème de la malléabilité, SegWit a ouvert la voie à d'autres développements en surcouche du système Bitcoin, notamment le réseau Lightning Network, qui permet des transactions rapides et à faible coût.

SEGWIT2X

Tentative controversée de hard fork visant à doubler la limite de taille des blocs sur Bitcoin, tout en intégrant SegWit. SegWit2x a été introduit lors du New York Agreement en 2017, une réunion confidentielle entre plus de 50 entreprises de l'écosystème qui visait à trouver une solution pour le passage à l'échelle du système. SegWit2x a cherché à augmenter la capacité transactionnelle de Bitcoin en portant la taille maximale d'un bloc à 2 Mo, contre 1 Mo initialement. Malgré le signalement positif de plus de 80 % des mineurs, le projet n'a pas réussi à obtenir un consensus, ce qui a mené à son annulation. Cet épisode a été perçu par beaucoup d'utilisateurs et développeurs comme une attaque contre Bitcoin.

SegWit2x est parfois également nommé « B2X » ou « S2X ». Initialement, son nom était « SegWit2Mb ».

SEGWIT V0

Version de script post-SegWit zéro. Les scripts SegWit V0 représentent la première famille de scripts introduite après la mise à jour SegWit de 2017. Les scripts `P2WPKH` et `P2WSH` incarnent la version SegWit V0. Les adresses correspondantes commencent toujours par `bc1q` et sont encodées en format `Bech32`.

SEGWIT V1

Version de script post-SegWit un. Les scripts SegWit V1 représentent la seconde famille de scripts introduite après la mise à jour SegWit de 2017. En l'occurrence, les scripts SegWit V1 ont été présenté avec la mise à jour Taproot en 2021. Le script `P2TR` est de la version SegWit V1. Les adresses correspondantes commencent toujours par `bc1p` et sont encodées en format `Bech32m`.

SÉLECTION DES PIÈCES

Processus par lequel un logiciel de portefeuille Bitcoin choisit quels UTXO utiliser comme entrées pour satisfaire les sorties d'une transaction. La méthode de sélection des pièces est importante, car elle a des impacts sur le coût d'une transaction et la confidentialité de l'utilisateur. Elle vise souvent à minimiser le nombre d'entrées utilisées, afin de réduire la taille de la transaction et les frais associés, tout en tentant de préserver la confidentialité en évitant de fusionner des pièces provenant de sources différentes (CIOH). Plusieurs méthodes existes pour la sélections de pièce comme le Knapsack Solver ou le Branch-and-Bound. Lorsque la sélection des pièces est réalisée manuellement par l'utilisateur, on parle alors de « Coin Control ».

En anglais, on parle de « Coin Selection ».

SELF-CUSTODY

Désigne la pratique par laquelle les utilisateurs gardent le contrôle direct de leurs clés privées, et donc de leurs bitcoins, sans dépendre d'une entité externe pour la gestion de ces actifs.

SELFISH MINING

Stratégie (ou attaque) dans le minage, où un mineur ou un groupe de mineurs conserve intentionnellement des blocs avec une preuve de travail valide sans les diffuser immédiatement sur le réseau. L'objectif est de conserver une avance sur les autres mineurs en termes de preuve de travail, ce qui leur permet potentiellement de miner plusieurs blocs d'affilée et de les publier en une seule fois, maximisant ainsi leurs gains. Autrement dit, le groupe de mineur attaquants ne minent pas sur le dernier bloc validé par l'ensemble du réseau, mais plutôt sur un bloc qu'ils ont eux-mêmes créé, qui diffère de celui validé par le réseau. Ce procédé génère une sorte d'embranchement secret de la blockchain, qui reste inconnue de l'ensemble du réseau jusqu'à ce que cette chaîne alternative dépasse potentiellement la blockchain honnête. Une fois que la chaîne secrète des mineurs attaquants devient plus longue (c'est-à-dire qu'elle contient plus de preuve de travail accumulé) que la chaîne honnête et publique, elle est alors diffusée sur l'ensemble du réseau. À ce moment, les nœuds du réseau, qui suivent la chaîne la plus longue (avec le plus de travail de preuve de travail accumulé), vont se synchroniser sur cette nouvelle chaîne. Il y a donc une réorganisation. Le selfish mining est embêtant car il diminue la sécurité du système en gaspillant une partie de la puissance de calcul du

réseau. En cas de réussite, il conduit également à des réorganisations de la blockchain, affectant ainsi la fiabilité des confirmations de transaction pour les utilisateurs. Cette pratique reste tout de même risquée pour le groupe de mineur attaquant, car il est souvent plus rentable de miner normalement au-dessus du dernier bloc connu publiquement plutôt que d'allouer de la puissance de calcul à un embranchement secret qui ne dépassera probablement jamais la blockchain honnête. Au plus le nombre de blocs dans la réorganisation est grand, au plus la probabilité de réussite de l'attaque est basse.

La traduction française de « selfish mining » est « minage égoïste ». Attention, une attaque par selfish mining ne doit pas être confondue avec une attaque de block withholding (bloc retenu).

SHA256

Sigle pour « Secure Hash Algorithm 256 bits ». C'est une fonction de hachage cryptographique produisant un condensat de 256 bits. Conçue par la *National Security Agency* (NSA) au début des années 2000, elle est devenue une norme fédérale pour le traitement des données sensibles. Dans le protocole Bitcoin, la fonction `SHA256` est omniprésente. Elle est employée pour hacher les entêtes des blocs dans le cadre de la preuve de travail. `SHA256` est également utilisée dans le processus de dérivation d'une adresse de réception à partir d'une clé publique. On l'utilise également pour l'agrégation des transactions et des témoins au sein des arbres de Merkle dans les blocs. On retrouve aussi `SHA256` dans le calcul d'empreinte de clés, le calcul de certaines sommes de contrôle et dans de nombreux autres processus autour de Bitcoin. Lorsqu'elle est appliquée deux fois de suite, on parle d'un `HASH256`. Cette double application est celle utilisée majoritairement sur Bitcoin. Lorsque `SHA256` est utilisé conjointement à la fonction `RIPEMD160`, on parle d'un `HASH160`. Ce double hachage est utilisé pour les empreintes de clés et pour le hachage de clés publiques. La fonction `SHA256` fait partie de la famille des SHA 2.

*Pour plus d'informations, voir la définition de **FONCTION DE HACHAGE CRYPTOGRAPHIQUE**.*

SHA512

Sigle pour « Secure Hash Algorithm 512 bits ». C'est une fonction de hachage cryptographique produisant un condensat de 512 bits. Elle a été conçue par la *National Security Agency* (NSA) au début des années 2000. Dans le protocole Bitcoin, la fonction `SHA512` est exclusivement utilisée dans le cadre des dérivations de clés enfants. Dans ce processus, elle est utilisée plusieurs fois dans l'algorithme `HMAC`, ainsi que dans la fonction de dérivation de clés `PBKDF2`. La fonction `SHA512` fait partie de la famille des SHA 2, comme `SHA256`. Son fonctionnement est d'ailleurs très similaire à cette dernière.

*Pour plus d'informations, voir la définition de **FONCTION DE HACHAGE CRYPTOGRAPHIQUE**.*

SHAREDCOIN

Service de mixage de pièces Bitcoin lancé en 2013 par Blockchain.info, mais qui n'est plus en service aujourd'hui. Ce service proposait aux utilisateurs d'améliorer leur confidentialité sur Bitcoin en combinant leurs transactions avec celles d'autres personnes, grâce à une technique de mixage similaire aux coinjoins. SharedCoin apportait une forme de confidentialité sans nécessiter de faire confiance au coordinateur, car les pièces des utilisateurs restaient sous leur contrôle tout au long du processus. Contrairement aux services de mixage centralisés de l'époque, les bitcoins ne pouvaient

pas être volés par l'intermédiaire. SharedCoin a par la suite fait face à des problèmes menant à la désanonymisassions de certains de leurs mixages au début de l'été 2014.

*Pour plus d'informations, voir la définition de **COINJOIN**.*

SHOR (ALGORITHME)

Algorithme quantique inventé en 1994 par Peter Shor permettant de factoriser des grands entiers en produit de nombres premiers en temps polynomial. En réduisant le nombre d'opérations nécessaires pour factoriser des entiers, Shor pourrait rendre impraticable les algorithmes de cryptographie établis sur ce problème mathématique comme RSA. Shor peut être légèrement modifié pour agir sur presque tous les algorithmes qui utilisent une structure de groupe. Il dispose notamment déjà d'une variante efficace sur la cryptographie sur les courbes elliptiques (ECDSA, Schnorr…). À l'heure actuelle, nous ne disposons pas encore d'un ordinateur quantique suffisamment puissant et stable pour exécuter avec succès l'algorithme de Shor. Shor et ses proches variantes sont donc efficaces sur les algorithmes de cryptographie asymétrique.

SIDECHAIN

Blockchain conçue pour fonctionner en parallèle avec la blockchain principale de Bitcoin. Les deux chaînes sont connectées à l'aide d'un ancrage bilatéral qui permet de faire en sorte que l'actif qui circule sur la sidechain conserve la même valeur que le bitcoin sur la chaîne principale. La sidechain dispose de son propre mécanisme de consensus qui peut être indépendant ou qui peut reposer en partie sur celui de la chaîne principale. Elle permet généralement d'utiliser des fonctionnalités qui ne sont pas disponibles directement sur la chaîne principale ou bien de bénéficier de fonctionnalités améliorées, comme par exemple : plus de flexibilité dans le développement, des transactions plus rapides et/ou plus confidentielles, ou encore, une capacité transactionnelle plus élevée. Pour ce faire, la sidechain fait des compromis par rapport à la chaîne principale. Ce concept de sidechain a initialement été présenté en 2014 par Adam Back, Matt Corallo, Luke Dashjr, Mark Friedenbach, Gregory Maxwell, Andrew Miller, Andrew Poelstra Jorge Timon et Pieter Wuille. Actuellement, les sidechains les plus connues sur Bitcoin sont Liquid et RSK (Rootstock). Ces dernières demeurent toutefois très peu utilisées par à d'autres solutions de surcouche avec un modèle différent comme le Lightning Network.

En français, on parle d'une « chaîne latérale » ou d'une « chaîne parallèle ».

SIGHASH_ALL (0X01)

Type de SigHash Flag utilisé dans les signatures des transactions Bitcoin pour indiquer que la signature s'applique à tous les composants de la transaction. En utilisant `SIGHASH_ALL`, le signataire couvre tous les inputs et tous les outputs. Cela signifie que ni les inputs ni les outputs ne peuvent être modifiés après la signature sans invalider celle-ci. Ce type de SigHash Flag est le plus courant dans les transactions Bitcoin, car il assure une finalité et une intégrité complètes de la transaction.

SIGHASH_ALL | SIGHASH_ANYONECANPAY (0X81)

Type de SigHash Flag combiné avec le modificateur `SIGHASH_ANYONECANPAY` utilisé dans les signatures des transactions Bitcoin. Cette combinaison spécifie que la signature s'applique à tous les outputs et uniquement à un input spécifique de la transaction. `SIGHASH_ALL | SIGHASH_ANYONECANPAY` permet à d'autres participants d'ajouter des inputs supplémentaires à la transaction après sa signature initiale. Elle est particulièrement utile dans des scénarios collaboratifs, comme les transactions

de financement participatif, où différents contributeurs peuvent ajouter leurs propres inputs tout en préservant l'intégrité des outputs engagés par le signataire initial.

SIGHASH_ANYPREVOUT

Proposition d'implémentation d'un nouveau SigHash Flag modificateur dans Bitcoin, introduite avec le BIP118. `SIGHASH_ANYPREVOUT` permet une plus grande flexibilité dans la gestion des transactions, en particulier pour des applications avancées comme les canaux de paiement sur le Lightning Network et la mise à jour Eltoo. Le `SIGHASH_ANYPREVOUT` permet de ne lier la signature à aucun UTXO spécifique antérieur (*Any Previous Output*). Utilisé en combinaison avec `SIGHASH_ALL`, il permettrait de signer tous les outputs d'une transaction, mais aucun input. Cela permettrait de réutiliser la signature pour différentes transactions, tant que certaines conditions spécifiées sont remplies.

> *Ce SigHash modificateur SIGHASH_ANYPREVOUT est hérité de l'idée du SIGHASH_NOINPUT initialement proposée par Joseph Poon en 2016 pour améliorer son idée du Lightning Network.*

SIGHASH_ANYPREVOUTANYSCRIPT

Variante du SigHash Flag modificateur `SIGHASH_ANYPREVOUT` dans Bitcoin. Ce SigHash fonctionne comme `SIGHASH_ANYPREVOUT`, mais il permet en plus de changer le script de sortie associé à un UTXO donné.

SIGHASH FLAG

Paramètre dans une transaction Bitcoin permettant de déterminer les composants d'une transaction (inputs et outputs) couvertes par la signature associée et deviennent donc immuables. Le SigHash Flag est un octet ajouté à la signature numérique de chaque entrée. Le choix du SigHash Flag affecte donc directement les parties de la transaction qui sont figées par la signature et celles qui peuvent encore être encore modifiées par la suite. Ce mécanisme assure que les signatures engagent les données de transaction de manière précise et sécurisée, selon l'intention du signataire. Trois principaux SigHash Flags existent :

-`SIGHASH_ALL` (`0x01`) : La signature s'applique à tous les inputs et outputs de la transaction, les verrouillant ainsi intégralement ;

-`SIGHASH_NONE` (`0x02`) : La signature s'applique à tous les inputs mais aucun output, permettant la modification des outputs après la signature ;

-`SIGHASH_SINGLE` (`0x03`) : La signature couvre tous les inputs et seulement un output correspondant à l'index de l'input signé.

En complément de ces trois SigHash Flags, le modificateur `SIGHASH_ANYONECANPAY` (`0x80`) peut être combiné avec chacun des types précédents. Quand ce modificateur est utilisé, seule une partie des inputs est signée, laissant les autres ouverts à modification. Voici les combinaisons existantes avec le modificateur :

-`SIGHASH_ALL | SIGHASH_ANYONECANPAY` (`0x81`) : La signature s'applique à un seul input tout en couvrant tous les outputs de la transaction ;

-`SIGHASH_NONE | SIGHASH_ANYONECANPAY` (`0x82`) : La signature couvre un seul input, sans engager aucun output ;

-`SIGHASH_SINGLE | SIGHASH_ANYONECANPAY` (`0x83`) : La signature s'applique à un seul input et uniquement à l'output ayant le même index que cet input.

Un synonyme parfois utilisé de « SigHash » est « Signature Hash Types ».

SIGHASH_NONE (0X02)

Type de SigHash Flag utilisé dans les signatures des transactions Bitcoin pour indiquer que la signature s'applique à tous les inputs de la transaction, mais à aucun de ses outputs. L'utilisation de `SIGHASH_NONE` implique que le signataire s'engage uniquement sur les entrées, mais permet que les sorties restent indéterminées ou modifiables après la signature. Ce type de signature est utile dans les cas où le signataire souhaite autoriser d'autres parties à décider de la manière dont les bitcoins seront distribués dans cette transaction.

SIGHASH_NONE | SIGHASH_ANYONECANPAY (0X82)

Type de SigHash Flag combiné avec le modificateur `SIGHASH_ANYONECANPAY` utilisé dans les signatures des transactions Bitcoin. Cette combinaison indique que la signature s'applique seulement à un input spécifique, sans engager aucun output. Cela permet aux autres participants de rajouter librement des inputs supplémentaires et de modifier tous les outputs de la transaction.

SIGHASH_SINGLE (0X03)

Type de SigHash Flag utilisé dans les signatures des transactions Bitcoin pour indiquer que la signature s'applique à tous les inputs de la transaction et à un seul output, correspondant à l'index du même input signé. Ainsi, chaque input signé avec `SIGHASH_SINGLE` est lié spécifiquement à un output particulier. Les autres outputs ne sont pas engagés par la signature et peuvent donc être modifiés ultérieurement. Ce type de signature est utile dans des transactions complexes, où les participants veulent lier certains inputs à des outputs spécifiques, tout en laissant de la flexibilité pour les autres outputs de la transaction.

SIGHASH_SINGLE | SIGHASH_ANYONECANPAY (0X83)

Type de SigHash Flag combiné avec le modificateur `SIGHASH_ANYONECANPAY` utilisé dans les signatures des transactions Bitcoin. Cette combinaison spécifie que la signature s'applique à un seul input spécifique et uniquement à l'output ayant le même index que cet input. Les autres inputs et outputs peuvent être ajoutés ou modifiés par d'autres parties. Cette configuration est utile pour des transactions collaboratives où les participants peuvent ajouter leurs propres inputs pour financer un output spécifique.

SIGNATURE NUMÉRIQUE

Preuve cryptographique qui démontre la possession d'une clé privée spécifique, associée à une clé publique unique, sans avoir à la divulguer. Sur Bitcoin, on la construit à l'aide de la clé privée et du hash d'une transaction. Elle atteste la propriété des bitcoins concernés et permet de satisfaire les conditions de dépense. Elle est générée grâce à un algorithme de signature numérique sur courbe elliptique tel qu'ECDSA ou le protocole de Schnorr.

SIGNET

Versions spécifiques du réseau Bitcoin conçues pour le développement et les tests. Les signets simulent le comportement du réseau principal (mainnet) mais avec la possibilité de contrôler divers

paramètres. Ils offrent ainsi un environnement pour tester de nouvelles fonctionnalités ou modifications sans risquer de perturber le réseau principal et sans en subir les frais. Par rapport au testnet, les signets offrent un contrôle plus structuré sur la génération de blocs, souvent géré par une ou plusieurs entités de confiance ou par un mécanisme de consensus personnalisé. Cela permet de créer des scénarios de test plus prévisibles, par rapport au testnet qui subit les aléas du minage, de la même manière que le mainnet.

SIGOPS (SIGNATURE OPERATIONS)

Désigne les opérations de signature numérique nécessaires pour valider les transactions. Chaque transaction Bitcoin peut contenir plusieurs inputs, chacun pouvant nécessiter une ou plusieurs signatures pour être considéré comme valide. La vérification de ces signatures se fait grâce à l'utilisation d'opcodes spécifiques que l'on nomme les « sigops ». Concrètement, cela inclut `OP_CHECKSIG`, `OP_CHECKSIGVERIFY`, `OP_CHECKMULTISIG` et `OP_CHECKMULTISIGVERIFY`. Ces opérations font peser une certaine charge de travail sur les nœuds du réseau qui doivent les vérifier. Pour éviter des attaques DoS par inflation artificielle du nombre de sigops, le protocole impose donc une limite sur le nombre de sigops autorisées par bloc, afin de garantir que la charge de validation reste gérable pour les nœuds. Cette limite est actuellement de 80 000 sigops maximum par bloc. Pour compter, les nœuds suivent les règles suivantes :

Dans le `scriptPubKey`, `OP_CHECKSIG` et `OP_CHECKSIGVERIFY` comptent pour 4 sigops. Les opcodes `OP_CHECKMULTISIG` et `OP_CHECKMULTISIGVERIFY` comptent pour 80 sigops. Lors du comptage, ces opérations sont en effet multipliées par 4 lorsqu'elles ne font pas partie d'un input SegWit (pour un P2WPKH, le nombre de sigops sera donc de 1) ;

Dans le `redeemScript`, les opcodes `OP_CHECKSIG` et `OP_CHECKSIGVERIFY` valent également 4 sigops, `OP_CHECKMULTISIG` et `OP_CHECKMULTISIGVERIFY` sont comptabilisés pour `4n` s'ils précèdent `OP_n`, ou 80 sigops dans le cas contraire ;

Pour le `witnessScript`, `OP_CHECKSIG` et `OP_CHECKSIGVERIFY` valent 1 sigop, `OP_CHECKMULTISIG` et `OP_CHECKMULTISIGVERIFY` sont comptés pour `n` s'ils sont introduits par `OP_n`, ou 20 sigops autrement.

Dans les scripts Taproot, les sigops sont traitées de manière différente par rapport aux scripts traditionnels. Au lieu de compter directement chaque opération de signature, Taproot introduit un budget de sigops pour chaque entrée de transaction, qui est proportionnel à la taille de cette entrée. Ce budget est de 50 sigops plus la taille en octets du témoin de l'input. Chaque opération de signature réduit ce budget de 50. Si l'exécution d'une opération de signature fait chuter le budget en dessous de zéro, le script est invalide. Cette méthode permet plus de flexibilité dans les scripts Taproot, tout en maintenant une protection contre les abus potentiels liés aux sigops, en les liant directement au poids de l'entrée. Ainsi, les scripts Taproot ne sont pas pris en compte dans la limite des 80 000 sigops par bloc.

SLIP (SATOSHI LABS IMPROVEMENT PROPOSALS)

Ensemble de propositions visant à améliorer ou à standardiser l'utilisation de Bitcoin, émanant de SatoshiLabs, la société à l'origine des portefeuilles matériels Trezor. Ces propositions s'articulent souvent comme des extensions de BIP (*Bitcoin Improvement Proposals*), dans le but d'enrichir les standards existants. Elles exposent les décisions techniques prises par SatoshiLabs qui ne trouvent pas leur place dans les BIP, mais qui restent pertinentes pour d'autres développeurs de logiciels de portefeuilles ou de portefeuilles matériels, notamment pour contribuer à l'uniformisation des processus.

SMPPS (SHARED MAXIMUM PAY PER SHARE)

Méthode de calcul de la rémunération des mineurs dans le contexte des pools de minage. C'est une variante de la méthode PPS. Elle limite les paiements de sorte que la pool ne paie jamais plus que ce qu'elle a gagné. Ainsi, même si les mineurs soumettent des parts valides, la récompense totale distribuée ne peut excéder les revenus de la pool. Cette méthode vise à maintenir l'équilibre financier de la pool tout en lissant les revenus des mineurs.

SOFT FORK

Modification des règles du protocole de manière rétrocompatible. Contrairement au hard fork, le soft fork ne donne pas lieu à une séparation du réseau de nœuds Bitcoin en deux groupes distincts, à condition qu'une majorité de la puissance de calcul se trouve sur la chaîne à jour. Si tout se passe bien, les nœuds avec la mise à jour et les nœuds sans la mise à jour restent donc sur la même blockchain. Une modification est dite rétrocompatible lorsqu'elle ajoute ou rend plus restrictives certaines règles du protocole.

SOMME DE CONTRÔLE (CHECKSUM)

La somme de contrôle est une valeur calculée à partir d'un ensemble de données, utilisée pour vérifier l'intégrité et la validité de ces données lors de leur transmission ou de leur stockage. Les algorithmes de somme de contrôle sont conçus pour détecter des erreurs accidentelles ou des altérations involontaires des données, comme les erreurs de transmission ou les corruptions de fichiers. Différents types d'algorithmes de somme de contrôle existent, tels que le contrôle de parité, les sommes de contrôle modulaires, les fonctions de hachage cryptographiques, ou encore les codes BCH (*Bose, Ray-Chaudhuri et Hocquenghem*). Dans le système Bitcoin, les sommes de contrôle sont employées pour assurer l'intégrité des adresses de réception. Une somme de contrôle est calculée à partir de la charge utile d'une adresse d'un utilisateur, puis ajoutée à cette adresse afin de détecter d'éventuelles erreurs lors de sa saisie. Une somme de contrôle est également présente dans les phrases de récupération (mnémonique).

> *La traduction anglaise de « somme de contrôle » est « checksum ». Il est généralement admis d'utiliser directement le terme de « checksum » en français.*

SOROBAN

Protocole de communication chiffré établi sur Tor permettant de collaborer avec d'autres utilisateurs dans le cadre d'une transaction Cahoots. Soroban a été développé par les équipes de Samourai Wallet afin de faciliter l'échange de transaction partiellement signées entre les utilisateurs qui souhaitent réaliser des transactions collaboratives (*Stowaway*, *Stonewall*, *StonewallX2*…). Ce protocole est utilisé sur l'application Samourai Wallet et sur le logiciel Sparrow Wallet.

SORTIE (OUTPUT)

Dans le contexte de Bitcoin, une « sortie » (ou « output » en anglais) au sein d'une transaction fait référence aux *Unspent Transaction Outputs* (UTXO) qui sont créés comme fonds de destination pour le paiement. Plus précisément, il s'agit d'un mécanisme par lequel une transaction distribue des fonds. Une transaction prend des UTXO, c'est-à-dire des morceaux de bitcoins, comme « inputs » (entrées) et crée de nouveaux UTXO comme « outputs » (sorties). Ces outputs stipulent une certaine quantité de bitcoins, souvent attribués à une adresse spécifique, ainsi que les conditions sous lesquelles ces fonds peuvent être dépensés ultérieurement. Le rôle de la transaction Bitcoin est donc de consommer des UTXO en entrées, et de créer des nouveaux UTXO en sorties. La

différence entre les deux correspond aux frais de transactions qui peuvent être récupérés par le mineur gagnant du bloc. Un UTXO est, en essence, la sortie d'une transaction précédente qui n'a pas encore été dépensée. Les outputs de transaction sont donc les créations de nouveaux UTXO qui seront, à leur tour, potentiellement utilisés comme inputs dans les transactions futures. D'un point de vue plus large, en informatique, le terme « output » ou « sortie » désigne généralement les données en résultat d'une fonction, d'un algorithme, ou d'un système. Par exemple, lorsque l'on passe une donnée dans une fonction de hachage cryptographique, cette information est nommée « entrée » ou « input », et le résultat est nommé « sortie » ou « output ».

SORTIE NON RENTABLE

Synonyme de « dust » ou « poussière » en français. Pour plus d'informations, voir la définition de **DUST**.

En anglais, on croise parfois le terme de « uneconomical outputs » pour désigner du dust.

SPEEDY TRIAL

Méthode d'activation de soft fork initialement conceptualisée pour Taproot début 2021 par David A. Harding sur une idée de Russell O'Connor. Son principe est d'utiliser la méthode du BIP8 avec un paramètre `LOT` réglé sur `faux`, tout en réduisant le délai d'activation à seulement 3 mois. Cette réduction du délai de vote permet une vérification rapide de l'approbation des mineurs. Si le seuil d'approbation requis est atteint pendant l'une des périodes, le soft fork est alors verrouillé. Il sera activé plusieurs mois plus tard, donnant ainsi aux mineurs le temps nécessaire pour mettre à jour leurs logiciels. Le succès de cette méthode pour Taproot, qui bénéficiait d'un large consensus au sein de la communauté Bitcoin, ne garantit cependant pas son efficacité pour toutes les mises à jour. Bien que la méthode Speedy Trial permette une activation plus rapide, certains développeurs expriment des inquiétudes quant à son utilisation future. Ils craignent qu'elle ne conduise à une succession trop rapide de soft forks, ce qui pourrait potentiellement menacer la stabilité et la sécurité du protocole Bitcoin. Par rapport au BIP8 avec le paramètre `LOT=true`, la méthode Speedy Trial est beaucoup moins menaçante envers les mineurs. Aucun UASF n'est prévu automatiquement. Cette méthode d'activation n'a pas encore été formalisée au sein d'un BIP.

« Speedy Trial » est emprunté d'une terminologie juridique qui indique un « procès expéditif ». Cela invoque le fait que la proposition d'amélioration est envoyée rapidement devant le tribunal des mineurs, afin d'être fixé sur leurs intensions. Il est généralement admis d'utiliser directement le terme anglais en français.

SPOF (POINT DE DÉFAILLANCE UNIQUE)

Un point de défaillance unique (SPOF, de l'anglais « Single Point of Failure ») désigne dans le domaine informatique un composant ou un élément d'un système dont la défaillance entraînerait l'arrêt complet ou une perte significative de fonctionnalités de l'ensemble du système. Il peut s'agir d'une pièce matérielle, d'une information, d'un logiciel, ou d'une partie d'un réseau. Par exemple, dans le contexte spécifique des portefeuilles HD Bitcoin, la phrase de récupération de 12 ou de 24 mots constitue souvent un SPOF pour le portefeuille. Si son secret n'est pas assuré, l'intégralité du portefeuille pourrait être subtilisé. De la même manière, sa simple perte pourrait entrainer la perte de l'intégralité des bitcoins du portefeuille.

SPREAD (WST)

Dans le logiciel Whirlpool Stat Tool, le spread est un indicateur permettant de mesurer l'homogénéité du processus de mixage du point de vue d'une pièce donnée. On différenciera 2 spread : le prospectif et le rétrospectif. Le spread prospectif est calculé en tant que ratio entre l'anonset prospectif de votre pièce et le nombre total de pièces créées après votre Tx0. Par exemple, si dans votre pool il y a 100 pièces et que votre pièce a un anonset de 70, le spread prospectif de votre pièce est alors de 70%. Le spread rétrospectif, quant à lui, est le ratio entre l'anonset rétrospectif de votre pièce et le nombre total de Tx0 créées avant le dernier mix de votre pièce. Ainsi, si l'anonset rétrospectif de votre pièce est de 95 et qu'il y a eu 100 Tx0 avant votre dernier mix, alors le spread rétrospectif de votre pièce est de 95%. Ces deux indicateurs permettent d'évaluer l'efficacité du mixage de votre pièce par rapport au potentiel offert par la pool. Un spread prospectif faible, comme 5% par exemple, indique une importante marge d'amélioration possible par des mixages supplémentaires. Inversement, un spread prospectif élevé, comme par exemple 97%, signifie que peu d'anonset supplémentaire peut être gagné.

> *En français, on pourrait traduire « spread » par « taux de diffusion » ou « taux de propagation ».*

STABLECOIN

Catégorie de cryptomonnaie conçue pour maintenir une valeur stable par rapport à un actif référence, souvent une monnaie fiduciaire comme le dollar américain.

STALE BLOCK

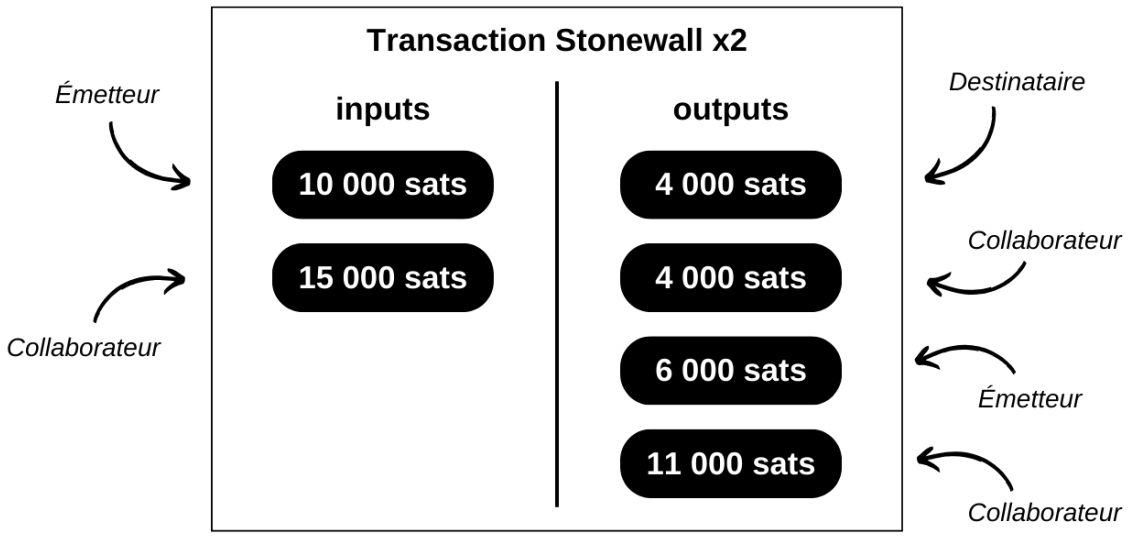
Fait référence à un bloc sans enfant (bloc obsolète) : un bloc valide mais exclu de la chaîne principale de Bitcoin.

> *Pour plus d'informations, voir la définition de **OBSOLÈTE (BLOC)**.*

STONEWALL X2

Forme spécifique de transaction Bitcoin visant à accroître la confidentialité des utilisateurs lors d'une dépense, par la collaboration avec une tierce personne non impliquée dans cette dépense. Cette méthode simule un mini-coinjoin entre deux participants, tout en effectuant un paiement à une troisième partie. Les transactions Stonewall x2 sont disponibles à la fois sur l'application Samourai Wallet et sur le logiciel Sparrow Wallet (les deux sont interopérables). Son fonctionnement est relativement simple : on utilise un UTXO en notre possession pour effectuer le paiement et on sollicite l'aide d'une tierce personne qui contribue également avec un UTXO lui appartenant. La transaction se solde avec quatre outputs : deux d'entre eux de montants égaux, l'un destiné à l'adresse du bénéficiaire du paiement, l'autre à une adresse appartenant au collaborateur. Un troisième UTXO est renvoyé à une autre adresse du collaborateur, lui permettant de récupérer le montant initial (une action neutre pour lui, modulo les frais de minage), et un dernier UTXO revient à une adresse nous appartenant, qui constitue le change du paiement. On définit ainsi trois rôles différents dans les transactions Stonewall x2 :

- L'émetteur, qui réalise le paiement effectif ;
- Le collaborateur, qui met des bitcoins à disposition afin d'améliorer l'ensemble d'anonymat de la transaction, tout en récupérant intégralement ses fonds à la fin ;
- Le destinataire, qui peut ignorer la nature spécifique de la transaction et attend simplement un paiement de la part de l'émetteur.



La structure Stonewall x2 ajoute énormément d'entropie à la transaction et vient brouiller les pistes de l'analyse de chaîne. Vue de l'extérieur, une telle transaction peut être interprétée comme un petit Coinjoin entre deux personnes. Mais en réalité, il s'agit d'un paiement. Cette méthode génère donc des incertitudes dans l'analyse de chaîne, voire oriente vers de fausses pistes. Même si l'observateur extérieur parvient à identifier le paterne de la transaction Stonewall x2, il ne disposera pas de toutes les informations. Il ne pourra pas déterminer lequel des deux UTXO de mêmes montants correspond au paiement. De plus, il ne sera pas en mesure de savoir qui a effectué le paiement. Enfin, il ne pourra pas déterminer si les deux UTXO en entrée proviennent de deux personnes différentes ou s'ils appartiennent à une seule personne qui les a fusionnés. Ce dernier point est dû au fait que les transactions Stonewall classiques suivent exactement le même paterne que les transactions Stonewall x2. Vu de l'extérieur et sans informations supplémentaires sur le contexte, il est impossible de différencier une transaction Stonewall d'une transaction Stonewall x2. Or, les premières ne sont pas des transactions collaboratives, alors que les secondes le sont. Cela permet d'ajouter encore plus de doutes sur la dépense.

STRATUM

Protocole réseau superposé à Bitcoin spécifiquement conçu pour optimiser la communication entre les mineurs individuels et les serveurs des pools de minage afin d'augmenter leur rentabilité. Stratum a été annoncé fin 2011 par Marek Palatinus, plus connu sous le pseudonyme de « Slush » et fondateur de la toute première pool de minage, Slush Pool, aujourd'hui rebaptisée Braiins. Stratum est venu remplacer l'ancien protocole Getwork, alors devenu obsolète. Il est important de comprendre que Stratum n'est pas intégré en tant que norme dans Bitcoin, mais est plutôt un logiciel spécifique utilisé par les pools. Bien que les pools de minage aient la liberté de ne pas l'utiliser, Stratum s'est imposé comme la référence pour le minage sur Bitcoin depuis plus de dix ans. Sa seconde version, Stratum V2, est actuellement en développement. Il vise à perfectionner Stratum et à réduire les inquiétudes de centralisation associées aux pools de minage que ce dernier a engendrées.

STRATUM V2

Évolution de Stratum, le célèbre protocole réseau superposé à Bitcoin conçu pour le minage. Ce type de protocole est conçu pour optimiser la communication entre les mineurs individuels et les serveurs des pools de minage afin d'augmenter leur rentabilité. Développé par la pool Braiins (anciennement « Slush Pool »), Stratum V2 introduit plusieurs améliorations, notamment une communication plus efficace entre les mineurs et les pools de minage, réduisant ainsi la bande passante nécessaire. Il réduit également les besoin d'infrastructures pour les pools. En termes de sécurité, il ajoute une authentification cryptographique afin d'empêcher différentes attaques de l'homme du milieu, notamment les écoutes clandestines et la redirection malveillante du hashrate. Un aspect important de Stratum V2 est qu'il inclut des sous-protocoles permettant aux mineurs individuels de choisir leurs propres ensembles de transactions à inclure dans les blocs sur lesquels ils travaillent. Cette fonctionnalité donne plus de pouvoir aux mineurs individuels, contrairement au protocole original où les pools avaient un contrôle total sur le bloc template. Cette méthode permettrait ainsi de renforcer la décentralisation du processus de minage par les pools.

SUBVENTION DE BLOC

Quantité de nouvelles unités pouvant être créées par le mineur qui résout un bloc. Cette subvention fait partie de la récompense de bloc avec les frais de transaction. Elle est distribuée au sein d'une transaction spécifique que l'on appelle « coinbase ». Initialement fixée à 50 bitcoins par bloc en 2009, cette subvention est réduite de moitié tous les 210 000 blocs (soit environ tous les quatre ans) grâce à un processus connu sous le nom de halving. Lorsque la subvention passera en dessous du montant de 1 sat, elle ne pourra plus être collectée, et la récompense de bloc reposera uniquement sur els frais de transaction. Sauf s'il y a une modification du protocole, la masse monétaire en circulation ne pourra plus être augmentée.

La traduction anglaise est « Block Subsidy ».

SURCOUCHE (LAYER)

Une « surcouche » (ou « layer » en anglais) est un protocole ou un réseau construit en supplément, en s'empilant sur le réseau Bitcoin principal. Elle utilise le réseau Bitcoin comme une fondation et est donc dépendante de son protocole. Cependant, le réseau Bitcoin n'est pas dépendant de ses surcouches. Un exemple d'une telle surcouche est le Lightning Network. Ces surcouches sont conçues pour étendre les capacités du réseau Bitcoin en ajoutant des fonctionnalités ou des capacités supplémentaires, telles que des transactions plus rapides, des jetons ou des micropaiements. Elles sont souvent créées pour résoudre certaines limitations du réseau Bitcoin, tout en bénéficiant de sa sécurité et de sa décentralisation. Il est important de noter que bien que ces surcouches soient construites sur le réseau Bitcoin, elles ont leurs propres protocoles et mécanismes distincts de ceux du réseau Bitcoin lui-même.

SYNCHRONISATION INITIALE D'UN NŒUD (IBD)

Traduction française de « *Initial Block Download* ». Fait référence au processus par lequel un nœud télécharge et vérifie la blockchain depuis le bloc Genesis, et se synchronise aux autres nœuds du réseau Bitcoin.

*Pour plus d'informations, voir la définition de **INITIAL BLOCK DOWNLOAD (IBD)**.*

T

TAPROOT

Mise à jour majeure du protocole Bitcoin, adoptée par le biais d'un soft fork en novembre 2021. Cette mise à jour apporte des améliorations significatives en termes de confidentialité, d'efficacité et de flexibilité. Elle permet l'utilisation du protocole de Schnorr et l'utilisation d'un script qui peut être révélé lors de la dépense. Le protocole de Schnorr, intégré à cette mise à jour, est un algorithme de signature numérique établi sur la cryptographie sur les courbes elliptiques (ECC), comme ECDSA. Dans le contexte de Bitcoin, Schnorr est utilisé pour générer une clé publique à partir d'une clé privée et pour signer une transaction avec une clé privée. Comme ECDSA sur Bitcoin, Schnorr utilise la courbe elliptique `secp256k1`, définie par l'équation $y^2 = x^3 + 7$. Les bitcoins bloqués avec Taproot peuvent être dépensés soit en satisfaisant l'un des scripts, soit en fournissant une signature valide correspondant à la clé publique, ce qui permet de garder les scripts privés. On y utilise un MAST pour permettre l'utilisation de plusieurs scripts. N'importe lequel peut-être utiliser pour dépenser les bitcoins associés. Cela permet des fonctionnalités plus complexes et des contrats intelligents plus sophistiqués.

*Pour plus d'informations, voir la définition de **SCHNORR (PROTOCOLE)**.*

TAPROOT ASSETS PROTOCOL

Protocole développé par Lightning Labs permettant d'émettre des actifs sur la blockchain principale de Bitcoin, en tirant partie de la mise à jour Taproot. Taproot Assets permet la création d'actifs fongibles comme des stablecoins et non fongibles comme de NFT. Les Taproot Assets peuvent être transférés via des transactions Bitcoin classiques ou via le Lightning Network. Ce protocole utilise des Merkle-Sum Sparse Merkle Trees (MS-SMT), une sorte de combinaison des MST et des SMT, pour assurer la validité et l'audibilité des actifs.

Taproot Assets Protocol s'appelait « TARO » auparavant.

TAPSCRIPT

Mise à jour qui a pour objet de modifier certains opcodes du langage de script classique de Bitcoin, afin de définir le nouveau langage de script utilisé pour les dépenses Taproot. Tapscript a été introduit par le BIP342 au sein du soft fork SegWit.

Afin de mettre en œuvre les diverses modifications associées à Taproot, il s'est avéré nécessaire de revisiter le langage de script. C'est là l'objet de Tapscript qui désactive ou modifie certains opcodes, et vient en ajouter de nouveaux.

*Pour plus d'informations, voir la définition de **SCHNORR (PROTOCOLE)** et **TAPROOT**.*

TARO

Ancien nom du protocole Taproot Assets Protocol.

*Pour plus d'informations, voir la définition de **TAPROOT ASSETS PROTOCOL**.*

TAUX DE HACHAGE

Traduction française de « *Hashrate* ».

*Pour plus d'informations, voir la définition de **HASHRATE**.*

TCP (TRANSMISSION CONTROL PROTOCOL)

Protocole de communication fondamental dans les réseaux, conçu pour assurer une transmission de données fiable sur Internet. Il établit une connexion, garantit l'ordre des données envoyées, gère la retransmission en cas de perte de paquets, et contrôle la congestion.

TÉMOIN DE TRANSACTION

Fait référence à une composante des transactions Bitcoin qui a été déplacée avec le soft fork SegWit afin de résoudre le problème de la malléabilité des transactions. Le témoin contient les signatures et les clés publiques nécessaires pour déverrouiller les bitcoins dépensés dans une transaction. Dans les transactions Legacy, le témoin représentait la somme des `ScriptSig` de tous les inputs. Dans les transactions SegWit, le témoin représente la somme des `ScriptWitness` de chaque input, et cette partie de la transaction est dorénavant déplacée dans un arbre de Merkle séparé au sein du bloc. Avant SegWit, les signatures pouvaient être légèrement modifiées sans être invalidées avant qu'une transaction ne soit confirmée, ce qui changeait l'identifiant de la transaction. Cela rendait difficile la construction de divers protocoles, car une transaction non confirmée pouvait voir son identifiant changer. En séparant les témoins, SegWit rend les transactions non malléables, car tout changement dans les signatures n'affecte plus l'identifiant de la transaction (TXID), mais uniquement l'identifiant du témoin (WTXID). En plus de résoudre le problème de la malléabilité, cette séparation permet d'augmenter la capacité de chaque bloc.

> *En anglais, « témoin » se traduit par « witness ». Pour plus d'informations, voir la définition de **SEGWIT**.*

TESTNET

Version alternative de Bitcoin utilisée exclusivement à des fins de test et de développement. Il s'agit d'un réseau séparé du réseau principal (mainnet), avec ses propres blocs et transactions, permettant aux développeurs de tester de nouvelles fonctionnalités, applications et mises à jour sans risque pour le réseau principal. Le testnet permet également d'éviter de payer des frais de transaction lors de tests. Les bitcoins utilisés sur le testnet n'ont aucune valeur réelle.

TIDES (TRANSPARENT INDEX OF DISTINCT EXTENDED SHARES)

Méthode de calcul de la rémunération des mineurs dans le contexte des pools de minage introduite par la pool OCEAN en 2023. Cette méthode répartit les récompenses en fonction d'un pourcentage pondéré du travail consacré aux preuves les plus récemment trouvées. Chaque preuve est rémunérée plusieurs fois, avec un calcul de récompense incluant les frais de transaction. Ce système assure une grande précision dans les paiements des mineurs, sans nécessiter un intermédiaire de garde pour le traitement des paiements, contrairement à d'autres méthodes comme FPPS. TIDES est conçu pour des rémunérations transparentes et auditables.

TIMELOCK

Primitive de contrat intelligent qui permet de définir une condition temporelle à remplir pour qu'une transaction puisse être ajoutée à un bloc. Il existe deux types de timelocks sur Bitcoin :

- Le timelock absolu, qui spécifie un moment précis avant lequel la transaction ne peut être incluse dans un bloc ;
- Le timelock relatif, qui définit un délai à partir de l'acceptation d'une transaction antérieure.

Le timelock peut être défini soit sous la forme d'une date exprimée en temps Unix, soit sous la forme d'un numéro de bloc. Enfin, le timelock peut s'appliquer soit à un output de transaction grâce à l'utilisation d'un opcode spécifique dans le script de verrouillage (`OP_CHECKLOCKTIMEVERIFY` ou `OP_CHECKSEQUENCEVERIFY`), soit à une transaction entière grâce à l'utilisation de champs de transaction spécifiques (`nLockTime` ou `nSequence`).

> *Pour plus d'informations, voir la définition de **OP_CHECKLOCKTIMEVERIFY**, **OP_CHECKSEQUENCEVERIFY**, **NLOCKTIME** et **NSEQUENCE**.*

TPRV

Préfixe de clé privée étendue pour les comptes Legacy et SegWit V1 sur Bitcoin Testnet.

> *Pour plus d'informations, voir la définition de **CLÉ ÉTENDUE**.*

TPUB

Préfixe de clé publique étendue pour les comptes Legacy et SegWit V1 sur Bitcoin Testnet.

> *Pour plus d'informations, voir la définition de **CLÉ ÉTENDUE**.*

TRANSACTION (TX)

Dans le contexte de Bitcoin, une transaction (abrégée « TX ») est une opération enregistrée sur la blockchain qui transfère la propriété de bitcoins d'une ou plusieurs entrées (inputs) vers une ou plusieurs sorties (outputs). Chaque transaction consomme des UTXO en entrées, qui sont des outputs de transactions précédentes, et crée de nouveaux UTXO en sorties, qui peuvent être utilisés comme entrants dans des transactions futures. Chaque entrée comporte une référence à un output existant ainsi qu'un script de signature qui rempli les conditions de dépense établies par l'output auquel il fait référence. C'est ce qui permet de débloquer des bitcoins. Les outputs définissent de nouvelles conditions de dépense pour les bitcoins transférés, souvent sous la forme d'une clé publique ou d'une adresse à laquelle les bitcoins sont maintenant associés.

TRANSACTION COINBASE

La transaction coinbase est une transaction spéciale et unique incluse dans chaque bloc de la blockchain Bitcoin. Elle représente la première transaction d'un bloc et est créée par le mineur qui a réussi à trouver un entête validant la preuve de travail (Proof-of-Work). La transaction coinbase sert principalement deux objectifs : attribuer la récompense de bloc au mineur et ajouter de nouvelles unités de bitcoins à la masse monétaire en circulation. La récompense de bloc, qui est l'incitation économique pour les mineurs à contribuer à s'adonner à la preuve de travail, comprend les frais accumulés pour les transactions incluses dans le bloc et un montant déterminé de bitcoins nouvellement créés ex-nihilo (subvention de bloc). Ce montant, initialement fixé à 50 bitcoins par bloc en 2009, est réduit de moitié tous les 210 000 blocs (environ tous les 4 ans) lors d'un événement appelé « halving ». La transaction coinbase diffère des transactions régulières de plusieurs manières. Tout d'abord, elle n'a pas d'entrée (input), ce qui signifie qu'aucune sortie de transaction existante (UTXO) n'y est dépensée. Ensuite, le script de signature `scriptSig` pour la transaction coinbase contient généralement un champ arbitraire permettant d'incorporer des données supplémentaires, telles que des messages personnalisés ou des informations de version de logiciel de minage. Enfin, les bitcoins générés par la transaction coinbase sont soumis à une période de maturité de 100 blocs (101 confirmations) avant de pouvoir être dépensés, afin de prévenir les dépenses potentielles de bitcoins non existants en cas de réorganisation de la chaîne.

TRANSACTION D'ENGAGEMENT

Dans le contexte d'un canal bidirectionnel au sein de Lightning, la transaction d'engagement est une transaction que les deux parties créent et signent, sans toutefois la publier sur la chaîne principale. Elle représente l'état actuel de la répartition des fonds entre les parties d'un canal, chaque paiement Lightning résultant en une nouvelle transaction d'engagement. Ces transactions sont valides, mais ne sont diffusées que lorsque le canal est clôturé unilatéralement. Elles contiennent des sorties pour chaque partie, reflétant la répartition des fonds selon les paiements Lightning effectués depuis l'ouverture du canal. Des mécanismes de pénalité sont associés pour dissuader les parties de diffuser des états obsolètes du canal, c'est-à-dire des vielles transactions d'engagement.

TUMBLEBIT

Concept de hub de paiement anonyme compatible avec Bitcoin proposé en 2016 par Ethan Heilman, Leen AlShenibr, Foteini Baldimtsi, Alessandra Scafuro et Sharon Goldberg. TumbleBit est un système de mixage de bitcoins qui ne requiert pas la confiance en un intermédiaire. Il permet à des utilisateurs de réaliser des paiements rapides, anonymes et hors-chaîne via un coordinateur appelé le Tumbler. TumbleBit garantit l'anonymat en s'assurant que même le Tumbler ne peut pas lier le paiement d'un payeur à son bénéficiaire. Le protocole TumbleBit assure que le Tumbler ne peut ni voler des bitcoins, ni imprimer de faux bitcoins en s'émettant des paiements à lui-même. L'anonymat offert par TumbleBit est comparable à celui d'un système eCash de Chaum. Cependant, ce concept n'a jamais été largement adopté, les techniques de confidentialité telles que le Chaumian Coinjoin lui étant préférées.

*Pour plus d'informations, voir la définition de **COINJOIN**.*

TWO-WAY PEG (2WP)

Traduction anglaise d'ancrage bilatéral.

*Pour plus d'informations, voir la définition d'**ANCRAGE BILATÉRAL**.*

TXID (TRANSACTION IDENTIFIER)

Identifiant unique associé à chaque transaction Bitcoin. Il est généré en calculant le hachage `SHA256d` des données de la transaction. Le TXID sert de référence pour retrouver une transaction spécifique au sein de la blockchain. Il est également utilisé pour pour faire référence à un UTXO, qui est essentiellement la concaténation du TXID d'une transaction précédente et de l'index de l'output désigné (également appelé « vout »). Pour les transaction post-SegWit, le TXID ne prend plus en compte le témoin de la transaction, ce qui permet de supprimer la malléabilité.

*Pour plus d'informations, voir la définition de **WTXID**.*

U

UASF (USER-ACTIVATED SOFT FORK)

Qualifie un soft fork dans Bitcoin lorsqu'il est initié et appliqué par les utilisateurs du réseau via leurs nœuds, sans dépendre de l'approbation des mineurs. Les nœuds du réseau mettent à jour leur logiciel pour adopter les nouvelles règles du soft fork et advienne que pourra. Typiquement utilisé en cas d'urgence, notamment lorsque les mineurs sont majoritairement opposés à l'adoption d'un soft fork, l'UASF sert de moyen de pression pour éviter une concentration excessive de pouvoir chez les mineurs. Dans les faits, l'UASF est même devenu un outil de dissuasion, agité par les opérateurs de nœuds lorsque les mineurs abusent de leur pouvoir. Toutefois, si l'UASF est réellement appliqué, il présente des risques, notamment la possibilité d'une scission de la blockchain, créant une nouvelle chaîne qui peut manquer de valeur économique et de sécurité. La première proposition formelle d'UASF provient du développeur Shaolin Fry, qui a poussé le BIP148 en mars 2017 pour faire pression sur les mineurs qui refusaient de signaler SegWit.

UDP (USER DATAGRAM PROTOCOL)

Protocole de communication utilisé sur Internet qui permet l'envoi de messages (datagrammes) entre ordinateurs sans établir de connexion préalable (contrairement à TCP). UDP est une méthode de transfert rapide mais sans garantie de livraison, d'ordre des paquets, ou de gestion d'erreur. On l'utilise plutôt pour des applications nécessitant une diffusion rapide et en temps réel. Ce protocole avait été utilisé au sein du projet FIBRE pour accélérer la propagation de blocs Bitcoin.

> *Pour plus d'informations, voir la définition de **FIBRE (FAST INTERNET BITCOIN RELAY ENGINE)**.*

UPRV

Préfixe de clé privée étendue pour les comptes Nested SegWit sur Bitcoin Testnet.

> *Pour plus d'informations, voir la définition de **CLÉ ÉTENDUE**.*

UPUB

Préfixe de clé publique étendue pour les comptes Nested SegWit sur Bitcoin Testnet.

> *Pour plus d'informations, voir la définition de **CLÉ ÉTENDUE**.*

URI (UNIFORM RESOURCE IDENTIFIER)

Format de chaîne de caractères standardisé utilisé pour identifier une ressource sur Internet. Un URI peut être soit un URL (*Uniform Resource Locator*), qui fournit un moyen d'accéder à une ressource en indiquant son emplacement sur un réseau informatique, soit un URN (*Uniform Resource Name*), qui nomme la ressource sans indiquer comment la localiser. Les URI sont centraux dans le fonctionnement du World Wide Web, car ils permettent d'accéder à des ressources comme des pages web, des documents et des services. Dans le contexte de Bitcoin, un URI est utilisé spécifiquement pour faciliter les transactions. Il permet d'encoder une adresse de réception, ainsi que d'autres paramètres d'une transaction comme le montant, dans un format standardisé selon le BIP21. Cela simplifie le processus de paiement en permettant aux utilisateurs de cliquer sur un lien ou de scanner un code QR, qui intègre automatiquement les informations nécessaires dans leur application de portefeuille Bitcoin.

UTXO

Sigle de *Unspent Transaction Output*. Un UTXO est une sortie de transaction qui n'a pas encore été dépensée ou utilisée comme entrée pour une nouvelle transaction. Les UTXOs représentent la fraction de bitcoins que possède un utilisateur et qui sont actuellement disponibles pour être dépensés. Chaque UTXO est associé à un script de sortie spécifique, qui définit les conditions nécessaires pour dépenser les bitcoins. Les transactions dans Bitcoin consomment ces UTXOs en entrées (inputs) et créent de nouveaux UTXOs en sorties (outputs). Le modèle d'UTXO est fondamental sur Bitcoin, car il permet de vérifier facilement que les transactions n'essaient pas de dépenser des bitcoins qui n'existent pas ou qui ont déjà été dépensés.

UTXO SET

Le terme « UTXO set » désigne l'ensemble de tous les UTXOs existants à un moment donné. Autrement dit, c'est une grosse liste de tous les différents morceaux de bitcoins qui attendent d'être dépensés. Si l'on additionne les montants de tous les UTXOs de l'UTXO set, cela nous donne la masse monétaire totale de bitcoins en circulation. Chaque nœud du réseau Bitcoin conserve son propre UTXO set en temps réel. Il l'actualise au fur et à mesure de la confirmation de nouveaux blocs valides, avec les transactions qu'ils incluent, qui consomment certains UTXOs de l'UTXO set, et qui en créent de nouveaux en contrepartie. Cet UTXO set est conservé par chaque nœud afin de pouvoir vérifier rapidement si les UTXOs dépensés dans les transactions sont bien légitimes. Cela leur permet de détecter et de rejeter les tentatives de doubles dépenses. L'UTXO set est souvent au cœur d'inquiétudes sur la décentralisation de Bitcoin, car sa taille augmente naturellement très rapidement. Puisqu'il faut en conserver une partie en RAM pour pouvoir procéder à la vérification des transactions en temps raisonnable, il est possible que l'UTXO set rende progressivement l'opération d'un nœud complet trop couteuse. L'UTXO set a également un fort impact sur l'IBD (Initial Block Download). Au plus on peut mettre une grande part de l'UTXO en RAM, au plus l'IBD est rapide. Sur Bitcoin Core, l'UTXO set est stocké dans le dossier nommé « chainstate ».

> *En français, on pourrait traduire « UTXO set » par « ensemble d'UTXO ». Pour plus d'informations, voir la définition d'**UTXO**.

V

VANITY (ADDRESS)

Adresse de réception personnalisée qui contient une séquence spécifique de caractères choisie par l'utilisateur, généralement pour des raisons esthétiques. Ces adresses sont générées en exécutant un processus de calcul, où de multiples clés privées sont créées jusqu'à ce que l'une d'entre elles corresponde à une adresse de réception contenant la séquence désirée. Ce processus ne compromet pas la sécurité de l'adresse, mais peut nécessiter un temps et des ressources de calcul considérables, surtout pour des séquences plus longues ou plus spécifiques. C'est une sorte de processus de brute force.

En français, on parle d'une « adresse personnalisée ».

VANITYGEN

Premier logiciel open source en ligne de commande utilisé pour créer des adresses de réception personnalisées (vanity address). Il fonctionne en générant et en testant par tâtonnement des paires de clés jusqu'à ce qu'une adresse de réception correspondant aux critères spécifiés par l'utilisateur (comme une certaine séquence de caractères spécifiques) soit trouvée. Vanitygen nécessite un processus de calcul intensif, particulièrement pour des séquences longues.

VPRV

Préfixe de clé privée étendue pour les comptes SegWit V0 sur Bitcoin Testnet.

*Pour plus d'informations, voir la définition de **CLÉ ÉTENDUE**.*

VPUB

Préfixe de clé publique étendue pour les comptes SegWit V0 sur Bitcoin Testnet.

*Pour plus d'informations, voir la définition de **CLÉ ÉTENDUE**.*

W

WABISABI

Protocole de coordination de CoinJoins utilisé sur le portefeuille Wasabi.

*Pour plus d'informations, voir la définition de **COINJOIN**.*

WALLET

Traduction anglaise de « portefeuille ».

*Pour plus d'informations, voir la définition de **PORTEFEUILLE**.*

WALLET.DAT

Fichier dans Bitcoin Core qui stocke des informations sur le portefeuille de l'utilisateur, telles que les clés privées et les transactions effectuées. Le fichier wallet.dat est chiffré pour assurer la sécurité des fonds. Depuis la version 0.16.0, ce fichier a été déplacé dans le dossier wallets.

WALLETS/DB.LOG

Fichier journal dans Bitcoin Core spécifique à la base de données des portefeuilles. Il enregistre les opérations et les événements liés à la base de données des portefeuilles pour la résolution de problèmes.

WALLET IMPORT FORMAT (WIF)

Méthode pour encoder une clé privée Bitcoin de manière à ce qu'elle puisse être importée ou exportée plus facilement entre différents portefeuilles. Le WIF est établi sur un encodage Base58Check, qui inclut des informations sur la version, la compression de la clé publique correspondante et une somme de contrôle pour la détection d'erreurs de saisie. Une clé privée WIF commence par les caractères `5` pour les clés non compressées, ou `K` et `L` pour les clés compressées, et contient tous les caractères nécessaires pour reconstituer la clé privée originale. Le format WIF fournit un moyen standardisé pour transférer une clé privée entre différents logiciels de portefeuille.

WASABI WALLET

Portefeuille Bitcoin axé sur la confidentialité offrant des fonctionnalités telles que le CoinJoin.

WATCHMEN

Dans le cadre de Liquid (sidechain de Bitcoin), ce sont des entités chargées de maintenir l'ancrage du L-BTC, le jeton natif de Liquid, en gérant et sécurisant les BTC utilisés en sous-jacent. Ils s'assurent que les actifs transférés entre la blockchain Bitcoin principale et la sidechain Liquid sont correctement verrouillés et débloqués. L'objectif de leurs actions est de maintenir la même valeur entre le L-BTC circulant sur Liquid et le BTC circulant sur Bitcoin. Dans Liquid, les watchmen font partie des fonctionnaires avec les blocksigners.

En français, on peut traduire « watchmen » par « gardiens ».

WATCH-ONLY WALLET

Un watch-only wallet (ou « portefeuille en lecture seule ») est un type de logiciel qui permet à un utilisateur de voir les transactions associées à une clé ou un ensemble de clés Bitcoin spécifiques, sans posséder les clés privées correspondantes. Il offre une visibilité sur le solde et l'historique des transactions, sans pour autant permettre de dépenser les fonds du portefeuilles. Par exemple, l'application Sentinel est un watch-only wallet.

WHIRLPOOL

Implémentation du protocole de coinjoins chaumiens ZeroLink, développée par les équipes du portefeuille Samourai Wallet. Whirlpool est actuellement disponible sur les portefeuilles Samourai Wallet (Android), Sparrow Wallet (PC) et Bitcoin Keeper (IOS et Android).

> *Pour plus d'informations, voir la définition de **COINJOIN** et de **ZEROLINK**.*

WHIRLPOOL STAT TOOL

Logiciel en ligne de commandes développé par Samourai Wallet qui permet de fournir les anonsets prospectifs et rétrospectifs d'une pièce mixée au sein de Whirlpool, ainsi que son taux de diffusion dans la pool. WST utilise l'algorithme HyperLogLogPlusPlus qui permet d'estimer le nombre de valeurs distinctes dans un très grand ensemble de données.

WITNESSSCRIPT

Script qui spécifie les conditions sous lesquelles les bitcoins peuvent être dépensés dans les UTXO P2WSH ou P2SH-P2WSH. Typiquement, les `WitnessScript` déterminent les conditions d'un portefeuille multisignatures sous standard SegWit. Dans ces standards de script, le `scriptPubKey` de l'UTXO (la sortie) contient un hachage du `WitnessScript`. Pour utiliser cet UTXO comme entrée dans une nouvelle transaction, le détenteur doit révéler le `WitnessScript` original, afin de prouver sa correspondance avec l'empreinte dans le `scriptPubKey`. Le `WitnessScript` doit alors être inclus dans le `ScriptWitness` de la transaction, qui contient également les éléments nécessaires pour valider le script, comme par exemple les signatures. Le `WitnessScript` est donc l'équivalent pour SegWit du `redeemScript` dans une transaction P2SH, à la différence près qu'il est placé dans le témoin de la transaction, et non dans le `ScriptSig`.

> *Attention, le WitnessScript ne doit pas être confondu avec le ScriptWitness. Tandis que le WitnessScript définit les conditions de dépense d'un UTXO P2WSH ou P2SH-P2WSH et constitue un script à part entière, le ScriptWitness contient les données de témoin de tout input SegWit.*

WTXID

Extension du TXID traditionnel, incluant les données de témoin (witness) introduites avec SegWit. Alors que le TXID est un hachage des données de transaction hors témoin, le WTXID est le `SHA256d` de l'intégralité des données de la transaction, témoin inclus. Les WTXID sont stockés dans un second arbre de Merkle dont la racine est mise dans la transaction coinbase. Cette séparation permet de supprimer la malléabilité du TXID de la transaction.

> *Pour plus d'informations, voir la définition de **TXID** et **SEGWIT**.*

X

XOR

Sigle de l'opération « Exclusive or », en français « Ou exclusif ». C'est une fonction fondamentale de la logique booléenne. Cette opération prend deux opérandes booléens, chacun étant `vrai` ou `faux`, et produit une sortie `vraie` uniquement lorsque les deux opérandes diffèrent. Autrement dit, la sortie de l'opération `XOR` est `vraie` si exactement un (mais pas les deux) des opérandes est `vrai`. Par exemple, l'opération `XOR` entre `1` et `0` donnera comme résultat `1`. Nous noterons : $1 \oplus 0 = 1$. De même, l'opération `XOR` peut être effectuée sur des séquences plus longues de bits. Par exemple, $10110 \oplus 01110 = 11000$. Chaque bit de la séquence est comparé à son homologue et l'opération `XOR` est appliquée. Voici la table de vérité de l'opération `XOR` :

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

L'opération `XOR` est utilisée dans de nombreux domaines de l'informatique, notamment dans la cryptographie, pour ses attributs intéressants comme :

- Sa commutativité : L'ordre des opérandes n'affecte pas le résultat. Pour deux variables D et E données : $D \oplus E = E \oplus D$;
- Son associativité : Le regroupement des opérandes n'affecte pas le résultat. Pour trois variables A, B et C données : $(A \oplus B) \oplus C = A \oplus (B \oplus C)$;
- Il dispose d'un élément neutre `0` : Une opérande xorée à 0 sera toujours égale à l'opérande. Pour une variable A donnée : $A \oplus 0 = A$;
- Chaque élément est son propre inverse. Pour une variable A donnée : $A \oplus A = 0$.

Dans le cadre de Bitcoin, on utilise évidement l'opération `XOR` à de nombreux endroits. Par exemple, le `XOR` est massivement utilisé dans la fonction `SHA256`, elle-même largement utilisée dans le protocole Bitcoin. Certains protocoles comme le *SeedXOR* de Coldcard utilisent également cette primitive pour d'autres applications. On le retrouve aussi dans le BIP47 pour chiffrer le code de paiement réutilisable lors de sa transmission. Dans le domaine plus général de la cryptographie, le `XOR` peut être utilisé tel quel comme un algorithme de chiffrement symétrique. On appelle cet algorithme le « Masque Jetable » ou le « Chiffre Vernam » du nom de son inventeur. Cet algorithme, bien qu'inutile en pratique du fait de la longueur de la clé, est un des seuls algorithmes de chiffrement reconnus comme inconditionnellement sûrs.

XPRV

Préfixe de clé privée étendue pour les comptes Legacy et SegWit V1 sur Bitcoin.

*Pour plus d'informations, voir la définition de **CLÉ ÉTENDUE**.*

XPUB

Préfixe de clé publique étendue pour les comptes Legacy et SegWit V1 sur Bitcoin.

*Pour plus d'informations, voir la définition de **CLÉ ÉTENDUE**.*

Y

YPRV

Préfixe de clé privée étendue pour les comptes Nested SegWit sur Bitcoin.

*Pour plus d'informations, voir la définition de **CLÉ ÉTENDUE**.*

YPUB

Préfixe de clé publique étendue pour les comptes Nested SegWit sur Bitcoin.

*Pour plus d'informations, voir la définition de **CLÉ ÉTENDUE**.*

Z

ZEROCONF

Pratique risquée consistant à considérer une transaction Bitcoin comme définitive, et à procéder à l'exécution de l'acte associé en contrepartie (tel que la vente d'un bien ou d'un service), avant que la transaction ne soit réellement incluse dans un bloc sur la blockchain. Les transactions non confirmées, ou en zeroconf, sont vulnérables à des attaques de double dépense, car elles ne sont pas encore irrévocablement inscrites dans le registre. Le zeroconf peut éventuellement être envisagé dans des contextes très spécifiques, où la rapidité est prioritaire, comme dans le cas de petites transactions commerciales ou dans le cas d'une transaction entre proches. Dans ces situations, le risque de double dépense est souvent considéré comme acceptable en comparaison de l'avantage d'une transaction rapide. Néanmoins, pour des transactions importantes, en particulier lorsqu'on ne connaît pas l'expéditeur, il est crucial d'attendre plusieurs confirmations avant de considérer la transaction comme immuable. La norme généralement acceptée est d'attendre 6 confirmations, ce qui signifie que 5 blocs supplémentaires doivent être minés après celui incluant la transaction, pour la considérer comme définitive.

ZEROLINK

Protocole de Chaumian Coinjoin qui vise à briser toutes les liaisons entre des ensembles de pièces séparées à travers des techniques avancées de mixage. Le protocole ZeroLink se distingue par sa capacité à protéger l'anonymat des utilisateurs contre diverses formes d'analyse de chaîne au niveau de la transaction et du réseau. ZeroLink introduit un cadre pour les portefeuilles de coinjoin, avec l'utilisation de comptes pré-mix et post-mix ségrégués, ainsi qu'une technique de mixage propre : le Chaumian Coinjoin. À ce jour, la seule implémentation de coinjoins ZeroLink est Whirlpool, disponible sur Samourai Wallet et Sparrow Wallet.

> *Pour plus d'informations, voir la définition de **CHAUMIAN COINJOIN**.*

ZEROSYNC

Projet développé pour tirer partie des ZKP (preuves à divulgation nulle de connaissance) dans l'écosystème Bitcoin, afin d'améliorer le passage à l'échelle et la confidentialité du système. Leur protocole principal (également nommé ZeroSync) permet de compresser l'historique de la blockchain Bitcoin, afin d'accélérer la synchronisation initiale des nœuds avec le réseau via l'utilisation d'une preuve compacte, sans pour autant compromettre la vérification.

ZKP (ZERO-KNOWLEDGE PROOF)

Méthode cryptographique permettant à une partie (le prouveur) de prouver à une autre partie (le vérificateur) qu'une information est vraie, sans révéler l'information ni aucun aspect de celle-ci. Une ZKP permet de garantir l'exactitude d'une affirmation, tout en préservant la confidentialité des données sous-jacente.

> *En français, on traduit Zero-Knowledge Proof par « Preuve à divulgation nulle de connaissance ».*

ZPRV

Préfixe de clé privée étendue pour les comptes SegWit V0 sur Bitcoin.

> *Pour plus d'informations, voir la définition de **CLÉ ÉTENDUE**.*

ZPUB

Préfixe de clé publique étendue pour les comptes SegWit V0 sur Bitcoin.

*Pour plus d'informations, voir la définition de **CLÉ ÉTENDUE**.*